On the Kidney Exchange Problem and Online Minimum Energy Scheduling

Tulia Herrera

Submitted in partial fulfillment of the

requirements for the degree

of Doctor of Philosophy

in the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2014

ABSTRACT

On the Kidney Exchange Problem and Online Minimum Energy Scheduling

Tulia Herrera

The allocation and management of scarce resources are of central importance in the design of policies to improve social well-being. This dissertation consists of three essays; the first two deals with the problem of allocating kidneys and the third one on power management in computing devices.

Kidney exchange programs are an attractive alternative for patients who need a kidney transplant and who have a willing, but medically incompatible, donor. A registry that keeps track of such patient-donor pairs can find matches through exchanges amongst such pairs. This results in a quicker transplant for the patients involved, and equally importantly, keeps such patients from the long wait list of patients without an intended donor. As of March 2014, there were at least 99,000 candidates waiting for a kidney transplant in the U.S. However, in 2013 only 16,893 transplants were conducted. This imbalance between supply and demand among other factors, has driven the development of multiple kidney exchange programs in the U.S. and the subsequent development of matching mechanisms to run the programs.

In the first essay we consider a matching problem arising in kidney exchanges between hospitals. Focusing on the case of two hospitals, we construct a strategy-proof matching mechanism that is guaranteed to return a matching that is at least 3/4 the size of a max-cardinality matching. It is known that no better performance is possible if one focuses on mechanisms that return a maximal matching, and so our mechanism is best possible within this natural class of mechanisms. For path-cycle graphs we construct a mechanism that returns a matching that is at least 4/5 the size of max-cardinality matching. This mechanism does not necessarily return a maximal matching. Finally, we construct a mechanism that is universally truthful on path-cycle graphs and whose performance is within 2/3 of optimal. Again, it is known that no better ratio is possible.

In most of the existing literature, mechanisms are typically evaluated by their overall performance on a large exchange pool, based on which conclusions and recommendations are drawn. In our second essay, we consider a dynamic framework to evaluate extensively used kidney exchange mechanisms. We conduct a simulation-based study of a dynamically evolving exchange pool during 9 years. Our results suggest that some of the features that are critical in a mechanism in the static setting have only a minor impact in its long-run performance when viewed in the dynamic setting. More importantly, features that are generally underestimated in the static setting turn to be relevant when we look at dynamically evolving exchange pool. For example, the pairs' arrival rates. In particular we provide insights into the effect on the waiting times and the probability to receive an offer of controllable features such as the frequency at which matching are run, the structures through which pairs could be matched (cycles or chains) as well as inherent features such as the pairs ABO-PRA characteristics, the availability of altruistic donors, and wether or not compatible pairs join the exchange etc. We evaluate the odds to receive an offer and the expected time to receive an offer for each ABO-PRA type of pairs in the model.

Power management in computing devices aims to minimize energy consumption to perform tasks, meanwhile keeping acceptable performance levels. A widely used power management strategy for devices, is to transit the devices and/or components to lower power consumption states during inactivity periods. Transitions between power states consume energy, thus, depending on such costs, it may be advantageous to stay in high power state during some inactivity periods. In our third essay we consider the problem of minimizing the total energy consumed by a 2-power state device, to process jobs that are sent over time by a *constrained adversary*. Jobs can be preempted, but deadlines need to be met. In this problem, an algorithm must decide when to schedule the jobs, as well as a sequence of power states, and the discrete time thresholds at which these states will be reached. We provide an online algorithm to minimize the energy consumption when the cost of a transition to the low power state is small enough. In this case, the problem of minimizing the energy consumption is equivalent to minimizing the total number of inactivity periods. We also provide an algorithm to minimize the energy consumption when it may be advantageous to stay in high power state during some inactivity periods. In both cases we provide upper bounds on the competitive ratio of our algorithms, and lower bounds on the competitive ratio of all online algorithms.

Table of Contents

List of Figures

List of Tables

To my beloved parents, sister and husband

Chapter 1

Introduction

1.1 On the kidney exchange problem

Kidney transplantation is the best known treatment for patients suffering from *end stage renal disease* (ESRD) [40]. As of March 2014, there were at least 99,000 candidates waiting for a kidney transplantation in the U.S. However, in 2013 only 16,893 transplants were conducted, of which 11,161 were carried-out with a deceased donor and 5,732 with a living donor[1]. This imbalance between supply and demand, together with the prevailing need to improve the quality of life and life expectancy of ESRD patients, and the substantial savings that performing a live kidney transplant generates, has driven the development, and subsequent expansion of multiple kidney exchange programs in the U.S. [20]. These programs are seen as an alternative to increase the pool of living donors.

In the last decade or so many countries have started kidney exchange programs as a way of combating the great and growing imbalance between supply and demand for kidneys. A kidney exchange program is a registry in which patients who need a kidney transplant are registered together with their generally incompatible willing donor. The idea is that two (or more) such patient-donor pairs may be able to exchange their donors, thereby utilizing the donor's kidneys, which would otherwise not be used at all. These exchanges also improve the status of the patients on the national wait list for kidneys, as patients who are able to get a transplant through a kidney

[1]According to SRTP/OPTN national data retrieved at http://optn.transplant.hrsa.gov/ on 28 March 2014

exchange will not join this wait list or will depart from it once they are transplanted. Donors without an intended recipient can also register, and several transplantation centers or hospitals may be affiliated to a kidney exchange program. In earlier stages of the development of the programs, exchanges were exclusively pairwise and surgeries occurred simultaneously, due to ethical issues and incentives. Later developments have promoted the implementation of simultaneous exchanges involving 3 or more pairs, and also exchanges initiated by a living donor without an intended recipient. A donor without an intended patient initiates a chain of exchanges in which transplants in the chain are not necessarily conducted simultaneously as in the case of cycles. Chains can terminate in a patient without an intended donor or could be left open to be continued as more pairs join the exchange program.

The compatibility between a potential recipient and a donor is mainly determined by their ABO[2] blood-types and by their tissue type compatibility. The tissue type incompatibility is determined by the presence in the patient's blood of antibodies to the donor's HLA (Human Leukocyte Antigen) profile. A failure in tissue type compatibility is also known as *positive crossmatch*.

The existence of multiple kidney exchange programs in the U.S. has driven innovation and development. However, it has also divided a national exchange pool, into smaller, thus limited, exchange pools. This has reduced the number of opportunities for patients with incompatible living donors. An analogous situation is encountered at transplantation centers. Depending on the matching mechanism used by the kidney exchange program of its affiliation, a transplantation center may have incentives to only report to the exchange program, the pairs that it could not match internally. A transplantation center may be one hospital or multiple hospitals under some sort of collaboration agreement. Incentives issues are also observed at the level of the patients, who may have multiple willing donors. There is by now a large literature on kidney exchange programs, starting with the pioneering work of Roth, Ünver and Sönmez in [33; 36; 35; 34] focused on incentives facing the patient-donor pairs in the system. With the growing number of kidney exchange programs, however, came the growing recognition that modeling the incentives of hospitals or transplantation centers is important as well. The strategic behavior of hospitals as agents of patient-donor pairs was first addressed by Ashlagi, Fischer, Kash and Procaccia in [7].

[2]ABO is the most important blood-group system in human-blood transfusion. According to it, blood-type can be classified as O, A, B, AB.

The innovation and development that have brought the kidney exchange programs are greatly evidenced in the evolution of matching algorithms designed to match pairs through cycles in large graphs. In [1], Abraham, Blum and Sandholm report that the U.S. market is projected to have around 8,000 participant pairs. This has motivated the design of algorithms to efficiently compute good matchings involving three or more patient-donor pairs [1; 27]. There is an ongoing debate about the trade-off between innovation and development driven by competition and the loss of efficiency inherent in a fractioned exchange pool. Although smaller pools are associated with losses in efficiency, they are also linked to faster decision making processes due to several factors such as fewer number of hospitals (centers), geographic distances, insurances etc. Understanding how the size of the pool affects the performance of an exchange program will provide insights into this debate.

Another important aspect of the kidney exchange market that has been widely studied is the type of structures that suffice to clear the exchange pool [3; 4; 31; 5]. Due to the inherent logistical constraints imposed by cycles- one surgery room need to be available for each patient and each donor in the cycle, it is crucial to understand if short exchanges suffice to achieve satisfactory levels of performance (a combination of fairness and efficiency); and if that is not the case, which strategies can help. It has been shown in that in large compatibility graphs, cycles with at most 3 pairs suffice to clear the exchange pool [4]. This result follows from the fact that in standard random graph models, when the number of nodes is large the graph tends to be highly connected as argued in [3]. Nonetheless, compatibility graphs that have been observed in reality are sparse and not large, and empirical results from available data show that matching longer chains results in a substantially better performance.

In the vast majority of the current literature the kidney exchange problem is studied in a static framework. Algorithms are typically evaluated by their performance on a large compatibility graph and conclusions and recommendations have been drawn based on such models. In reality, pairs arrive over time, matching algorithms are run at different points in time, and the compatibility graph just before running a matching not only depends on the pairs that have arrived since the last run, but also on the pairs that weren't matched in the previous runs. Different strategies may benefit/hurt different type of pairs over time, and those outcomes determine the size and the PRA-ABO composition of the compatibility graph over time. Data-based dynamic evaluations have been

conducted by Dickerson et al. [10; 16; 18; 17]. Ünver [39] considered a dynamic model in which pairs arrive following a Poisson process. He designed an optimal dynamic policy for a model in which pairs do not have tissue type incompatibilities, there are no departures without a transplant and that there are arbitrarily many under demanded pairs in the exchange pool in the long run. He also shows that in his model, it is not always an optimal strategy to match as many pairs as possible at a given point in time.

Our work In Chapter 2 we continue the line of research that calls for designing mechanisms that encourage transplantation centers to report *all* their registered pairs to the kidney exchange program. For the transplantation centers to feel comfortable about reporting all their pairs, a conservative approach is to design a *strategy-proof* mechanism: in such a mechanism, it is a dominant strategy for the agents to truthfully report all the patients registered with them[3]. Specifically we consider the problem of designing a *strategy-proof* mechanism for a centralized planner with two agents that maximizes the total number of pairs that are matched through pairwise exchanges. Each pair is represented by only one of the two agents and the goal of each agent is to maximize the number of its patients matched by the exchange. We emphasize that this model does not capture the dynamic nature of the kidney exchange problem.

Following Ashlagi, Fischer, Kash and Procaccia in [8] and Caragiannis and Filos-Ratsikas [15], we consider the case of two agents. For general graphs we construct a strategyproof matching mechanism that is guaranteed to return a matching that is at least 3/4 the size of a max-cardinality matching. No better performance is possible if one focuses on mechanisms that return a maximal matching; our mechanism always yields a maximal matching, and so our mechanism is best possible within this natural class of mechanisms. This answers an open question of Caragiannis et al. For path-cycle graphs we construct a randomized, truthful in expectation mechanism that returns a matching that is at least 4/5 the size of a max-cardinality matching. This mechanism does not necessarily return a maximal matching (A path-cycle graph is a graph in which each component is either a path or a cycle). Finally, we construct a randomized mechanism that is universally truthful on path-cycle graphs that is at least 2/3 the size of a max-cardinality matching. Again, it is known that no better ratio is possible.

[3] As is common in this literature, we assume that monetary compensations are disallowed.

In Chapter 3 we propose a dynamic framework to evaluate extensively used kidney exchanges strategies. We do this through a simulation-based study in which parameters are estimated from data. In our model there is a single kidney registry at which pairs register directly, and to which they arrive and depart over time. Pairs are classified by the donor's blood type, the patient's blood type and their sensitization level. We consider two main scenarios depending on whether or not compatible pairs join the exchange program. For each scenario we simulate the exchange program during 9 years and study the effect of the (i) frequency at which matchings are run; (ii) presence of altruistic donors; (iii) maximum length of cycles allowed in clearing the exchange pool; and (iv) sensitization level of arriving patients on the (i) odds of the various types of pairs to receive an offer (ii) waiting time distribution (iii) sensitization level of the pool over time (iv) benefit brought by the presence of altruistic donors; and (v) bridge donor's blood type distribution. Instead of providing overall performance measures, we analyze the effects on each pair type. Due to the large disparity in the raw numbers among the different types of pairs, overall measures could potentially hide large disparity in the performance across different pair types, thus it is important to evaluate the performance of policies in a more detailed way instead of aggregating the performance measured across the various types.

We use as benchmark the scenario that results from matching at different frequencies the maximum number of pairs that could be matched through arbitrary length cycles. We propose policies that use exchanges of length at most 5 including chains in the presence of altruistic donors. Our findings suggest that while overall chains and long cycles improve efficiency, they also create large disparity in the odds to receive an offer and waiting time distribution across the different type of pairs. The frequency at which matchings are run does not have a significant effect on the number of offers that are made when compatible pairs join the pool for any of the scenarios we consider, however it has a substantial effect when only incompatible pairs join the pool. In this latter case the frequency at which matchings are run impacts differently the different type of pairs. There is a trade off in efficiency and waiting times across the different type of pairs when matching more or less frequency which should be considered when evaluating a policy. The role of a pair donor's blood type or patient's blood type or its sensitization level varies largely depending on the distribution of the flow of arriving pairs — for example when compatible pairs do not enter the exchange our findings suggest that the donor's blood type plays a central role in the probability of a patient

to receive an offer however when compatible pairs do not enter the pool this central role relies on the patient's blood type. On the presence of altruistic donors, particularly when 1% of the arrivals are altruistic donors, we find that the increase in the number of offers when chains are initiated is roughly 4%, however when compatible pairs do not join the pool this drops to 1%, and in both cases the improvement on highly sensitized patients seems to be minor. Finally regarding the distribution of the last bridge donor of a chain, not surprisingly, the brigde donor's blood type distribution changes significantly with respect to the altruistic donor's blood type distribution that initiated a chain. For example, while 65% of the altuistic donors in our model have blood type O, the percentage of blood type O bridge donors that terminate chains is below 15%. Measures across the different type of pairs as well as the interaction with the waiting list should be more carefully incorporated in the evaluation and design of policies for the kidney exchange. In particular, it is easy to overestimate the benefits obtained from redirecting altruistic donors from the waiting list to start chains in the exchange pool, the effect of this on the patients on the waiting list should be carefully evaluated.

1.2 Online scheduling for energy minimization

Energy conservation is a central concern in modern world. In 1992, the U.S. Environmental Protection Agency started Energy Star[4] to promote energy-efficiency in electronic devices. Several federal and state programs have been launched to promote the development of energy-friendly technologies as well as to promote the use of cleaner energy sources. In computing devices, energy consumption is a major concern. The power consumption rates of computing devices have increased exponentially bringing with it many technological, economical, and environmental challenges. For example, the amount of energy consumed worldwide by data centers in 2010 accounted for between 1.1% and 1.5% of the total electricity use; and in the U.S. it accounted for between 1.7% and 2.2% of total energy consumption, see [26]. As reported in [13], Google enginners warned that if energy consumption continues to grow, energy costs can outweigh hardware costs by a large margin.

This issue has been addressed from different perspectives such as hardware design, system de-

[4] "ENERGY STAR is a U.S. Environmental Protection Agency voluntary program that helps businesses and individuals save money and protect our climate through superior energy efficiency." https://www.energystar.gov/

sign, algorithmic techniques etc. From the algorithmic perspective, three main type of energy saving strategies have been widely considered: power-down strategies, speed scaling and combinations of both. Power-down strategies save energy by transitioning a device to a lower power state when there is certain level of inactivity — anybody with a battery-operated electronic device is familiar with this type of mechanism to conserve energy. Speed-scaling strategies can be implemented in variable-speed processors. They make use of the full spectrum of speeds at which the processor can operate and decide at which speed the processor should be operating at every point in time as to achieve the desired level of performance — for example meeting deadlines. The dynamic speed scaling problem without the sleep state was first studied by Yao, Demers and Shenker [42]. Irani and Pruhs [24] and Albers [2] survey the extensive literature on dynamic speed scaling strategies in the offline and in the online setting.

In a power-down strategy, the device always resides in one of several power states and transitions across states have a fixed cost. In the most simple version of this problem there are two power states. The device consumes H energy units per unit of time at high power state, L energy units per unit of time at low power state, and U energy units to move from low power state to high power state. Notice that is is assumed that the energy consumed to transit from the high power state to the low power state is zero. This assumption can be made because the device is assumed to be at the high power state when the inactivity period starts and terminates at the high power state when the inactivity period is over, then the number of transitions from low to high power state is equal to the number of transitions from high to low and thus the energy consumed in the two directions can be included in one quantity. The two-state version of the problem corresponds to a continuous version of the well-known *ski rental* problem [21]. It is generally assumed that when the device is processing, it does so at the maximum power rate, which is generally called the active state. The question that arises in this setting is when to transit and to which state should the device transit during inactivity periods so as to minimize the total energy consumed during an inactivity period. As the length of the inactivity period is unknown, a we would like to design online strategies to manage power during inactivity periods with provable performance guarantee with respect to an optimal solution that knows in advance the length of the inactivity period. For this problem, the *competitive ratio* of an online algorithm ALG is defined as an upper bound on the ratio between the total energy consumed by algorithm ALG during the inactivity period and

the energy consumed by an optimal offline algorithm that knows the length of the idle period in advance, over all possible lengths for the inactive period.

The problem previously described is well understood. Karlin, Manasse, McGeoch and Owicki showed [25] that in the two-power state case an optimal deterministic algorithm is 2-competitive and that an optimal randomized algorithm is $\frac{e}{e-1}$-competitive, in both cases they provided an algorithm with the corresponding performance guarantee. Later on, Irani, Shukla and Gupta studied [23] the case in which there are multiple power states. They presented online and an offline algorithm under some assumptions on the energy required to transit among different states. They provided a 2-competitive deterministic algorithm when the device has multiple power states, which is the best competitive ratio a deterministic algorithm can achieve. Also they generalized the result in the randomized scenario when the device has multiple power states. Finally, Augustine, Irani and Swamy [9] provide a deterministic algorithm that for arbitrary energy parameters produces a deterministic strategy whose competitive ratio is arbitrarily close to optimal ratio and provide a simple $3+2\sqrt{(2)} \approx 5.828$-competitive strategy. Notice that this problem does not model scheduling decisions, and is just concerned about power management during inactivity periods.

In many real life applications, the scheduler decides when and which job to schedule as well as in which power-state the device should operate. In Chapter 4, we study an online version of the problem of scheduling a set of jobs with release times and deadlines on a two-state processor so as to minimize the total energy consumed. We assume that jobs are processed in the high power state and the scheduler decides when to process the jobs and how to manage the power during the inactivity periods. The corresponding offline problem with objective to minimizing the total number of idle periods, was shown to be polynomial-time solvable by Baptiste [11]. When the switching energy costs are sufficiently small, minimizing the total number of idle periods is equivalent to minimizing the total energy consumption. When switching costs are larger, the problem is more challenging. For this case, Baptiste, Chrobak and Durr [12] provide a polynomial time algorithm to minimize the total energy consumption.

In many real-life problems the existence of jobs and their characteristics are not known until they are released, hence the online version of this problem is of special interest. Unfortunately, any online algorithm needs to process jobs as soon as they can be processed in order to guarantee that deadlines will be met. To see this, consider the following example: a job with workload 1

and deadline 2 is released at time 0; if this job is not processed at time 0, then the adversary can release a job with deadline 2 and workload 1, thus at least one of them will not be able to meet its deadline. This implies that online algorithms for this problem have no scheduling decisions to make, and hence, they have no control on the number or length of idle periods. In many situations, standard worst-case analysis may be pessimistic. For example, we may know that our system is unlikely to be so heavily loaded that jobs must run as soon as possible. To model this, we introduce a *constrained adversary*, who is limited in how much workload may be placed on the machine. The constrained adversary has two parameters, $s_{\max}$ and $\rho_{\max}$. The jobs release by the constrained adversary have span at most $s_{\max}$ and for an arbitrary time t $\sum_{j:r_j \leq t < d_j} \frac{\omega_j}{s_j} - \omega_j$ is the workload of job j. We consider the problem of minimizing the total energy consumed by a two-power state device, to process jobs that are sent over time by a *constrained adversary*. Jobs can be preempted, but deadlines need to be met. In this problem, an algorithm must decide when to schedule the jobs, as well as at which power state it should run at any time.

We provide an online algorithm to minimize the energy consumption for arbitrary values of power consumption rates. We also provide an online algorithm to minimize the total number of idle periods in the resulting schedule. In both cases, we provide upper bounds on the competitive ratio of our algorithms, and lower bounds on all online algorithms. The bounds presented hold for arbitrary power consumption rates of the processor.

The energy required to process an instance is at least the energy necessary to process the workload. Given that the machine has a constant power consumption rate when processing, it follows that the differences between schedules arises from two main features: the number and length of the idle periods in the schedules, and the sequence of power states followed by the algorithm during the idle periods. To solve the main problem addressed in this paper, we propose an algorithm - called DEFER - that approaches these two main features independently. In other words, scheduling decisions are taken independently of the power consumption rates. Decisions regarding when to process jobs and which jobs to process are taken using our algorithm LAZY. Algorithm LAZY waits as long as it is possible to start processing the pending workload as to guarantee that all deadlines will be met, this includes the deadline of pending jobs as well as the deadlines of any future possible arrival. Once it starts processing, LAZY processes under earliest deadline first until there is no more pending workload. Decisions regarding the sequence of power

states followed by the algorithm during the idle periods are taken using an optimal competitive deterministic algorithm. During an idle period, an optimal competitive deterministic algorithm stays in high power consumption state until the energy that it has consumed during that idle period is equal to the energy that it would have consumed if it had switched at the start of the idle period plus the energy required to power-up. We argue in this work that this separation is appropriate. The jobs' span are bounded by a parameter $s_{\max}$ and bounds are given as a function of $s_{\max}$. We show that for an arbitrary online algorithm the competitive ratio has an asymptotic lower bound in $\Omega(\ln s_{\max})$. We also show that DEFER has an asymptotic upper bound in $O\left(s_{\max}^{1/3}\right)$.

Chapter 2

An Optimal Randomized Mechanism for Kidney Exchange with Two Agents

2.1 Introduction

Patient-donor pairs can register in a kidney exchange program directly or through a transplantation center that is a member of the program. In the latter case, a transplantation center acts as an *agent* of its patient-donor pairs. In an ideal scenario, each center reports all their registered pairs to the exchange program, but it is not completely clear if this will actually be the case. If the kidney exchange programs cannot guarantee a reasonable number of matches for the patients registered through a given center, then that center may have an incentive to *not* reveal everyone who is registered with them. As an example, if a transplantation center can match more of its pairs by not reporting all of its pairs to the exchange program and instead matching some of them internally, then it is reasonable to expect that this center will actually do so. Often this may come at the cost of the total number of exchanges that can be done. As a way of addressing this drawback, research in this area has focused on understanding the efficiency loss of natural mechanisms as well as on designing mechanisms where this worst-case efficiency loss is as small as possible.

In this paper we continue this line of research that calls for designing mechanisms that encourage

transplantation centers to report all its registered pairs to the kidney exchange program. For the transplantation centers to feel comfortable about reporting all their pairs, a conservative approach is to design a *strategy-proof* mechanism: in such a mechanism, it is a dominant strategy for the agents to truthfully report all the patients registered with them.[1]Specifically we consider the problem of designing a *strategy-proof* mechanism for a centralized planner who wishes to maximize the total number of pairs matched through pairwise exchanges, assuming there are exactly two agents. Each pair is represented by only one of the two agents and the goal of each agent is to maximize the number of *its* patients matched by the exchange.

There is by now a large literature on kidney exchange programs, starting with the pioneering work of Roth et al. [32]. To the extent that the early literature focused on incentives, it focused on incentives facing the *patients* in the system, see Roth et al. [32; 36; 31]. With the growing number of kidney exchange programs, however, came the growing recognition that modeling the incentives of hospitals or transplantation centers is important as well.

Model and Prior Results. The incentive problem faced by the transplantation centers when only pairwise exchanges are allowed can be modeled as a matching problem on a graph. Each node in a graph represents a patient-donor pair; an edge connecting two nodes i and j indicates that i and j can participate in a feasible exchange. Each node of the graph is controlled by exactly one agent, which is the transplantation center where this pair is registered. By a mechanism, we mean a function that returns a matching for any input graph. Given a mechanism $\mathcal{M}$ and a graph G reported to the centralized planner, the utility of an agent r is simply the total number of nodes controlled by r that are matched by $\mathcal{M}$ on reported graph G, plus the total number of nodes controlled by agent r that are matched internally using nodes not reported to the central planner and those left unmatched by $\mathcal{M}$. Each agent wishes to maximize his utility. We assume that the set of nodes controlled by an agent is private information, so that the only nodes that become known to the mechanism are those that are reported by the agent. We also assume that the agents have the capability to conduct internal matches by themselves, meaning a transplantation center is able to match its own registered pairs without revealing them to the planner. We assume there is a central planner who wishes to maximize the total number of patients who receive kidneys, i.e., who wishes

[1] As is common in this literature, we assume that monetary compensations are disallowed.

to find a matching of maximum size. However, if the planner implements this maximum-matching mechanism, the transplantation centers may have an incentive to not report all of their registered patients. The central question studied here is: what is the worst-case loss in efficiency if we are required to use a mechanism in which the transplantation centers have no incentive to lie? We comment briefly on the metrics used to measure efficiency loss and strategyproofness. We measure the approximation ratio of a mechanism by the worst-case ratio of the maximum number of nodes that can be matched to the number of nodes matched by the mechanism. The worst-case is taken over all possible instances of the problem, and the goal is to design a mechanism for which the approximation ratio is as close to 1 as possible. For deterministic mechanisms, truthfulness (or stategy-proofness) is unambiguous as we demand this property in the dominant strategy sense. For randomized mechanisms, however, there are (at least) two notions of truthfulness. A randomized mechanism is *universally truthful* if it is a probability distribution over deterministic mechanisms, each of which is truthful. A randomized mechanism is *truthful in expectation* if the *expected* benefit for an agent from misreporting is non-positive. Obviously, a universally truthful mechanism is also truthful in expectation, but not vice-versa. We also classify mechanisms according to the maximality of the returned matching. A mechanism is inclusion-maximal if it always return a maximal matching[2]. That the efficient mechanism—the one that always finds a maximum number of matches—can be manipulated was recognized very early by Sönmez and Ünver [37]. Motivated by this observation, Ashlagi et al. [6] present MIX-AND-MATCH—a universally truthful randomized 2-approximation algorithm for an arbitrary number of agents. This algorithm is based on a simple deterministic 2-approximation algorithm that is truthful. Subsequently, Ashlagi and Roth [5] and Toulis and Parkes [38] study similar problems in random graphs that are modeled based on realistic data. For these problems they construct compelling mechanisms that satisfy *individual rationality*. In their MIX-AND-MATCH paper, Ashlagi et al. [6] proposed a mechanism called FLIP-AND-MATCH for the case of two agents. They showed that it is a 4/3-approximation, but were not able to prove that it was truthful in expectation. This question was settled by Caragiannis et al. in [15] who showed that FLIP-AND-MATCH is not strategy-proof in expectation. Given the difficulty in understanding the case of many agents, Caragiannis et al. focused on the two-agent case and proposed a mechanism with compelling properties. Their mechanism—WEIGHT-AND-

[2] A matching M is maximal if $M \cup \{e\}$ is not a matching for every $e \notin M$

MATCH—first assigns a weight to the edges of the graph in a specific way, and then they randomly choose between two maximum-weight matchings: the one with the fewest number of edges and the one with the most number of edges. It turns out that the max-weight matching with the fewest number of edges is a strategyproof mechanism, but the one with the most number of edges is not. However, they show that mixing the two gives an algorithm that is truthful in expectation and has an approximation ratio of 3/2 (and that this is tight). In addition, they improved several lower-bounds on approximation ratios: they showed a general lower bound of 5/4 on all mechanisms, 4/3 on all *inclusion-maximal* mechanisms, if the mechanism needs to be truthful in expectation; for universally truthful mechanisms, they strengthened these bounds to 3/2 and 2 respectively. These bounds all apply to problems with two agents (and hence also to more than two agents).

Our Contributions. Following Caragiannis et al. [15], we consider the case of two agents. For general graphs we construct a strategyproof matching mechanism that is guaranteed to return a matching that is at least 3/4 the size of a max-cardinality matching. As mentioned earlier, no better performance is possible if one focuses on mechanisms that return a maximal matching; our mechanism always yields a maximal matching, and so our mechanism is best possible within this natural class of mechanisms. This answers an open question of Caragiannis et al. For path-cycle graphs[3] we construct a randomized, truthful in expectation mechanism that returns a matching that is at least 4/5 the size of a max-cardinality matching. This mechanism does not necessarily return a maximal matching. Finally, we construct a randomized mechanism that is universally truthful on path-cycle graphs that is at least 2/3 the size of a max-cardinality matching. Again, it is known that no better ratio is possible.

2.2 Algorithm `BALANCE` and its analysis

We let R denote the set of nodes controlled by the gray agent and B the set of nodes controlled by the black agent. All the edges in the undirected graph—those connecting nodes of the same agent and those connecting nodes of different agents—are commonly observable. We let $G[R]$ denote the graph induced by the gray nodes; $G[B]$ denotes the graph induced by the black nodes; and

[3] A path-cycle graph is a graph in which each component is either a path or a cycle

$G := G[R \cup B]$ denotes the entire graph.

We set forth some basic notation that will be useful in the rest of the paper. A matching M in the graph G is simply a subset of the edges of G such that no two edges in the subset are incident to the same node. A node v is *matched* if it is adjacent to an edge in the matching, and is *free* otherwise; an edge e is matched if $e \in M$. Note that whether a node (or edge) is matched or free is always with respect to a matching M, so when M changes the status of the nodes may change as well. A path P in the graph G is a sequence $(v_1, v_2, \ldots, v_k)$ of nodes of G such that (v_i, v_{i+1}) is an edge of G for $i = 1, 2, \ldots, k-1$. Often it will be convenient to think of the path P as the sequence (or even subset) of the edges that are in P. To keep things simple, we use P to denote both the path itself and the subset of its edges; the interpretation we have in mind will usually be clear from the context. A path is an *alternating* path with respect to M if its edges (in sequence) alternate between matched and unmatched edges. An alternating path with respect to M is *augmenting* if its first and last edges are unmatched; equivalently, an alternating path is augmenting if its first and last nodes are free. By a *black augmenting* path, we mean an augmenting path whose two end-points are controlled by the black agent; similar definitions apply to the terms such as a *gray* augmenting path, a black alternating path, a gray-black augmenting path, etc. Note that we do not need to distinguish between a gray-black augmenting path and a black-gray one, because the graph is undirected. A node v in the graph $G = (N, E)$ is *vital* if the size of the maximum cardinality matching of graph $G' = (N \setminus \{v\}, E)$ is strictly less than the size of a maximum cardinality matching of G.

If M is a matching and P is an augmenting path with respect to M, then define

$$M \oplus P := (M \cup P) - (M \cap P).$$

Note that $M \oplus P$ is the matching obtained by switching the matched and free edges of P (and by retaining every edge of M that does not appear in P). We say that $M \oplus P$ is the matching obtained by augmenting M using the path P. Note that $M \oplus P$ is a matching and matches one extra edge (or two extra nodes, which are the endpoints of P) relative to M; moreover, every matched node in M continues to be a matched node in $M \oplus P$. A well-known result due to Berge shows that M is a maximum-cardinality matching if and only if there is no augmenting path with respect to M.

We first propose a strategy-proof deterministic mechanism `BALANCE` that is near-optimal. Specifically, we show that `BALANCE` finds a maximal matching and hence matches at least half the number

of nodes that can be matched by a maximum-cardinality matching. Note that it is easy to construct a deterministic truthful mechanism that guarantees a 2-approximation. The construction of `BALANCE` is (surprisingly) somewhat intricate. Our main goal in constructing `BALANCE` is to use it to design a more efficient randomized mechanism, and for this task we make use of a key property `BALANCE` that may not necessarily hold in other 2-approximations. It appears that this additional structure is necessary. Instead of specifying the algorithm in pseudo-code, we include a detailed description of its various steps. At the end of each step, the resulting graph will satisfy certain properties, and it is convenient to note these properties soon after describing the corresponding step.

Given R and B, the algorithm `BALANCE` in $G[R \cup B]$ works as follows:

1. Let M_i^B and M_i^R be *any* maximum cardinality matching of the graphs $G[B]$ and $G[R]$ respectively. Denote the total number of nodes matched in M_i^B and M_i^R as B_i and R_i respectively. Let $M = M_i^B \cup M_i^R$.

Note that at the end of step 1, all vital nodes are matched by M.

2. If P is a gray-black augmenting path with respect to M, augment M using the path P (equivalently set $M \leftarrow M \oplus P$); repeat this step until we find a matching M that has no gray-black augmenting path.

Note that at the end of step 2, every augmenting path is either gray or black. Also, because there is no gray-black augmenting path with respect to M, any gray augmenting path cannot have a common edge neither a common node with any black augmenting path. None of the subsequent steps of the algorithm will create of a gray-black augmenting path.

3. If there is a gray augmenting path P_R with respect to M and a black augmenting path P_B with respect to M, augment M using the paths P_R and P_B; repeat this step until we find a matching M for which every augmenting path (if any) is of one color.

The only augmenting paths (if any) with respect to the matching M at the end of step 3 are those involving exactly one of the two agents. Without loss of generality, we take that agent to be the black agent. The rest of the description of the algorithm is based on this assumption.

4. Let U_R be the set of free nodes of agent R (with respect to the "current" matching M), and let V_B be the set of *non-vital* black nodes reachable from some node in U_R via an *even-length alternating* path with respect to M. By assumption, every alternating path starting from a node in U_R must start with a free edge and end with a matched edge (otherwise it will be an augmenting path). If there is a black augmenting path P_{B} with respect to M, and if $V_B \neq \emptyset$, then augment M using the path P_B and update the result by switching the matched and free edges in some path from $r \in U_R$ to $b \in V_B$. Repeat this step as long as there is a black augmenting path and $V_B \neq \emptyset$. (Note that this step cannot create any augmenting paths involving a free gray node.)

This completes the description of the algorithm. It is clear that the algorithm terminates: every iteration of steps 2 and 3 results in at least one extra matched edge; in addition, each iteration of step 4 decreases the size of U_R by 1, and V_B must be empty whenever U_R is.
Define the *score* of an agent in a matching M to be the number of his nodes matched in M and let $\mu_B(M)$ and $\mu_R(M)$ denote the *score* of the black and the gray agent respectively. The main idea behind the algorithm is the following. We first start with the maximum score that the gray and black agents can achieve for themselves if they are isolated (i.e., if the other agent is absent). We then try to increase their scores by the same amount: every iteration of step 2 increases each of their scores by 1; and every iteration of step 3 increases each of their scores by 2. In each iteration of step 4, the black score goes up by 2 because of the augmentation, but drops by 1 because we switch the matched and free edges in the alternating path starting from a free gray node and ending with a matched black node; effectively, this leads to an increase of 1 in both the black and gray scores. We illustrate how the algorithm works on a simple example that appears in the Appendix (Figures 1-5).

Proposition 2.2.1. *At every iteration of Step 4 of* `BALANCE`*, the number of vertex-disjoint augmenting paths strictly decreases or* `BALANCE` *terminates.*

Proof. Let G be an arbitrary graph, and let $f(G)$ be the output of `BALANCE` when applied on G. Assume WLOG that at termination of Step 3, there are no augmenting paths with a gray leaf. By definition of `BALANCE`, at each iteration of Step 4 a path is alternated and another path is augmented or the algorithm stops. This implies that the size of the matching strictly increases

during each iteration of Step 4 or `BALANCE` terminates. From the relationship between the size of the matching and the number of disjoint augmenting paths, we can conclude that the number of disjoint augmenting paths strictly decreases or `BALANCE` terminates.

□

From Proposition 2.2.1 we can conclude that that once a node of agent black is unmatched during execution of Step 4, it will never be matched again. To see this, let b be a black node that was unmatched during the execution of Step 4 and later matched again. Consider the iteration of Step 4 such that at its termination the unmatched node b is the leaf of an augmenting path. During the execution of this iteration a path involving a gray node r is alternated and a path involving two black nodes is augmented. However the fact that at termination there is an augmenting path that involves node b implies that at the beginning of the iteration there is an augmenting path that involves node r. Given that at termination of Step 3 there is no augmenting path with a gray leaf, it follows that such path was created during the execution of Step 4, which contradicts the fact that at each iteration of Step 4 the number of vertex-disjoint augmenting paths strictly decreases.

Proposition 2.2.2. *Let $f(G)$ be the output of the mechanism* BALANCE *on the graph G. Then either $f(G)$ is a maximum-cardinality matching, or it maximizes the number of matched nodes for at least one of the agents subject to the constraint that all vital nodes are matched. In other words, $f(G)$ maximizes the total score, or the gray score or the black score while matching all vital nodes.*

Proof. Let M be the matching at the end of step 3 of the algorithm (i.e., just before step 4 is executed). We already know that every M-augmenting path is of one color, and without loss of generality, we take this to be black. The algorithm terminates after executing step 4 as many times as needed, updating M in each of those steps. Note that for the algorithm to terminate, either $U_R = \emptyset$; or $V_B = \emptyset$; or the matching M has no black augmenting paths. We show that in the last case, the algorithm finds a max-cardinality matching, whereas in the other two cases the algorithm maximizes the number of gray nodes that are matched subject to matching all vital nodes. If $U_R = \emptyset$, every gray node is matched, so the gray score is maximized trivially. Similarly, if there is no black augmenting path with respect to M, then there is no augmenting path with respect to M (from Proposition 2.2.1 it follows that no augmenting path is created in the last step), and so M is a max-cardinality matching.

The only remaining case to consider is when $V_B = \emptyset$, but $U_R \neq \emptyset$. We wish to argue that M maximizes the gray score subject to matching all vital nodes. Let M' be a matching that maximizes the gray score subject to matching all vital nodes (and if there are many such matchings, pick one with maximum cardinality, breaking ties arbitrarily). We argue that the gray score in M is exactly the same as that in M'. Consider the components of the symmetric difference of M and M'. It is well known that each component in the symmetric difference is either an even-length cycle, or a path (of odd or even length).

- In an even-length cycle both M and M' match all the nodes, and hence the same number of gray nodes.
- In an even-length path, the end nodes cannot be vital nodes as they are unmatched in M or in M'. Suppose the leaf nodes belong to different agents; If the gray leaf node is not matched by M, there is an even-length alternating path from an unmatched gray node to a black non vital node, contradicting $V_B = \emptyset$; if the gray leaf node is matched by M, then flipping the edges of M and M' only in this component, we can find a matching M'' that matches more gray nodes than M' does while matching all vital nodes , contradicting the definition of M'. Finally when both leaf nodes belong to the same agent, both M and M' match the same number of gray nodes.
- In an odd-length path, if the first and last edges are matched in M, both M and M' match the same number of gray nodes (and the leaf nodes of the path have to be black). If the first and last edges are matched in M', then the two leaf nodes of the path have to be black as well: otherwise there is an M-augmenting path with a gray leaf, which we assume does not exist. Again, M and M' match the same number of gray nodes.

Thus we find that M matches the same number of gray nodes as M'. As M' maximizes the gray score subject to matching all vital nodes, so does M.

□

Note that if M is the outcome of BALANCE, then there is a $\delta \geq 0$ such that $\mu_R(M) = R_i + \delta$ and $\mu_B(M) = B_i + \delta$. In other words, the "gain" over the best that each agent can do individually

is the *same* for both the gray and black agents in BALANCE. (This also justifies the name BALANCE for the algorithm.) We next prove that BALANCE cannot be manipulated by either agent.

Proposition 2.2.3. *Let $f(G)$ be the output of the algorithm* BALANCE *on the graph G. If v is unmatched in $f(G)$ then the score of the agents in $f(G\backslash\{v\})$ is the same as their score in $f(G)$.*

Proof. Let G be an arbitrary graph and let $G' = G\backslash\{v\}$. Let us assume without loss of generality that v is a *black* node and that it is unmatched in $f(G)$. Suppose the black agent's score is larger in $f(G)$ than in $f(G\backslash\{v\})$. Because all the vital nodes are matched at termination of `BALANCE` on any graph, it follows that $\delta > \delta'$ because $B_i = B'_i$ and $R_i = R'_i$. But this contradicts Proposition 2.2.2 because $f(G')$ is not a maximum cardinality matching and does not match for any agent the maximum number of nodes that it could match subject to matching all vital nodes. The latter follows from the fact that $f(G)$ is a matching on G' that matches more nodes of both agents while matching all vital nodes. □

From Proposition 2.2.3 it follows that no agent can increase its score by reporting a node that is not going to be matched in $f(G)$. Thus, we can assume that the score of an agent is given by its score in the matching returned by the centralized planner plus its score in a maximum cardinality matching of the graph induced by the nodes not reported by the agent.

Next we prove a result that establishes a relationship between a set of vital nodes and the number of nodes matched in a maximum cardinality matching of a graph in which such vital nodes are forbidden to be matched.

Proposition 2.2.4. *Let V be a set of vital nodes of the graph $G = (N, E)$. The number of nodes matched in a maximum cardinality matching of the graph induced by the nodes in $N\backslash V$ is at most $\mu(G) - |V|$.*

Proof. Let M be an arbitrary maximum cardinality matching of G. Let G' be the graph induced by the nodes in $N\backslash V$. Let M' be a maximum cardinality matching in G'. Clearly, M' is also a matching of G. As none of the nodes in V are matched in M', it follows that

$$|M| \quad \geq \quad |M'| + |V|,$$

which is the desired result.

□

Now we proceed to prove strategy-proofness and the approximation ratio of `BALANCE`.

Proposition 2.2.5. *Mechanism* BALANCE *is truthful.*

Proof. Let R_e be the maximum gray score when all the vital nodes in the graph are matched (over all matchings); similarly, let B_e be the maximum black score when all vital nodes in the graph are matched (over all matchings). For any matching M, we let $\mu_R(M)$ and $\mu_B(M)$ be, respectively, the number of gray and black nodes matched in M. For any graph G, we let $\mu(G)$ be the number of nodes matched by in a maximum cardinality matching of graph G. Recall that R_i (resp. B_i) denotes the maximum number of nodes matched by the gray (resp. black) agent in the subgraph induced by the gray (resp. black) nodes alone. Proposition 2.2.2 implies that the outcome of BALANCE is a maximum-cardinality matching, or one that maximizes the gray score, or one that maximizes the black score subject to matching all vital nodes. We prove that BALANCE is strategyproof in each case.

Suppose BALANCE finds a maximum cardinality matching $f(G)$ on the graph G. Then $\mu_R(f(G)) = R_i+\delta$ and $\mu_B(f(G)) = B_i+\delta$, for some $\delta \leq \min\{R_e-R_i, B_e-B_i\}$. Suppose the black agent "hides" a subset of his nodes so that the H is the subgraph induced by the hidden nodes (known only to the black agent), let $\{G_k\}$ be the set of graphs seen by the mechanism and let F be the subgraph induced by the hidden nodes and by the nodes of the black agent that are not matched in $f(\cup_k G_k)$. From Proposition 2.2.3 it follows that no agent have incentive to report nodes that will remain unmatched to increase its score in $f(G)$, so we can assume without loss of generality that $|\mu(H)| = |\mu(F)|$. For the black agent to do strictly better with this strategy, the gray agent must do strictly worse as BALANCE found a max-cardinality matching on G. Thus, we must have $\mu_R(f(\cup_k G_k)) < \mu_R(f(G))$. But by the key property of BALANCE, $\mu_R(f(\cup_k G^k)) = \sum_k R_i^k + \delta' = R_i + \delta'$, for some $\delta' \geq 0$. This implies that $\delta' < \delta$. Now, the number of black nodes matched when black hides is simply $2 \cdot |\mu(F)| + \mu_B(f(\cup_k G^k))$, and we have:

$$\begin{aligned} |\mu(F)| + \mu_B(f(\cup_k G^k)) &= |\mu(H)| + \sum_k B_i^k + \delta' \\ &\leq B_i + \delta' < B_i + \delta = \mu_B(f(G)). \end{aligned}$$

Thus, it is not possible for the black agent to do better by hiding any portion of his subgraph in this case. A a similar argument applies for the gray agent.

Suppose BALANCE maximizes the score of one of the agents subject to matching all vital nodes. From the definition of `BALANCE` it follows that $\mu_B(f(G)) = B_i + \delta$, $\mu_B(f(G)) = R_i + \delta$ and $\delta = \min\{R_e - R_i, B_e - B_i\}$. Without loss of generality, we assume that it maximizes the score of the gray agent, and so $\delta = R_e - R_i$. Suppose that the gray agent hides a subset of his nodes so that H is the corresponding induced subgraph, $\{G_k\}$ is the set of graphs seen by the mechanism and F is the graph induced by nodes in H and gray nodes not matched in $f(\cup_k G_k)$. Let us define the matching $\bar{M}$ as the union of matching $f(\cup_k G_k)$ and a maximum cardinality matching of F. From the definition of `BALANCE`, all the vital nodes of agent B in G are matched by $\bar{M}$. Consider the symmetric difference between $\bar{M}$ and M'.

- In an even length cycle, $\bar{M}$ and $f(G)$ matched the same nodes and so the same number of gray nodes.
- In an odd length path, we should only consider the case in which the path starts and finishes with an edge in $\bar{M}$. In this case both leaves are black nodes, otherwise there is an augmenting path respect to $f(G)$ with a gray leaf, which is a contradiction. Thus, in these components $\bar{M}$ and $f(G)$ matches the same number of gray nodes.
- In an even length path, if the path has leaves of the same color, then both matchings match the same number of gray nodes in these components. Suppose the path has a black leaf and a gray leaf; If the black leaf is matched by $f(G)$, then it follows that this leaf node is a vital node of agent B, otherwise one extra gray node could be matched by alternating such path without unmatching a vital node, contradicting the definition of R_e. But in this case, $\bar{M}$ does not match all the vital nodes of agent B, thus we can assume that the black leaf is matched by $\bar{M}$ and then it follows that the gray leaf is matched by $f(G)$ and so, $f(G)$ matches more gray nodes than $\bar{M}$ in these components.

We have shown that $f(G)$ matches at least as many gray nodes as $\bar{M}$, in other words, the gray agent has no incentive to hide nodes.

Suppose the black agent hides a subset of his nodes so that H is the induced subgraph, let $\{G_k\}$ the set of graphs seen by the mechanism and let F be the graph induced by the nodes in H and the black nodes that are left unmatched in $f(\cup_k G_k)$. From the key property of `BALANCE`, the black and gray score in the manipulated graph are $\sum_k B_i^k + \mu(F) + \delta'$ and $\sum_k R_i^k + \delta' = R_i + \delta'$

respectively. If $\delta' \leq \delta$, then it follows trivially that agent black has no incentive to be untruthful because $\mu(F) + \sum_k B_i^k \leq B_i$. Let us assume that $\delta' = \delta + \gamma$ for some $\gamma > 0$. Let M' be the union of the output of `BALANCE` on G', and a maximum cardinality matching of F. We will first argue that:

$$\sum_k B_i^k + \mu(F) \quad \leq \quad B_i - \gamma.$$

Let us consider the symmetric difference between M' and $f(G)$. The definition of `BALANCE` together with assumption that there are only black augmenting paths respect to $f(G)$ imply that the only components in the symmetric difference between M' and $f(G)$ in which M' matches more gray nodes than $f(G)$ are even length paths with one gray leaf and one black leaf in which the gray leaf is matched by M' and the black leaf is matched by $f(G)$. This follows because all odd length paths have black leaves, otherwise it contradicts the assumption that there are only black-black augmenting paths respect to $f(G)$. And in all even length cycles both M' and $f(G)$ match all nodes.

We can assume without loss of generality that in the symmetric difference between M' and $f(G)$ there are exactly γ even length path components with a gray leaf matched by M' and a black leaf matched by $f(G)$. Let us denote such paths as $P_1, \ldots, P_\gamma$. Let P_j be an arbitrary path among paths $P_1, \ldots, P_\gamma$; denote the gray leaf as r_j, and the black leaf as b_j. From definition of `BALANCE` it follows that b_j is a vital black node in G. Otherwise, during execution of Step 4, `BALANCE` could have alternated P_j to unmatch b_j and match r_j, while augmenting a black augmenting path. The fact that b_j is not matched by M' implies that b_j is not a vital node, in the graph induced by the black nodes in G', or in the graph F.

Let us consider the problem of finding a maximum cardinality matching in the graph induced by black nodes in G subject to forbidding matching the nodes $b_1, \ldots, b_\gamma$. From Proposition 2.2.4 it follows that:

$$\mu\left(G \backslash \{b_1, \ldots, b_\gamma\}\right) \quad \leq \quad B_i - \gamma. \tag{2.2.1}$$

From the definition of G' and F together with the fact that nodes $b_1, \ldots, b_\gamma$ are not matched in M', it follows that $\sum_k B_i^k + \mu(F) \leq \mu\left(G \backslash \{b_1, \ldots, b_\gamma\}\right)$. This together with inequality 2.2.1 imply

that:

$$\sum_k B_i^k + \mu(F) \quad \leq \quad B_i - \gamma. \tag{2.2.2}$$

Given that BALANCE matches $\delta + \gamma$ black nodes in Step 2 through Step 4 in G', the black agent utility is:

$$\begin{aligned} \left(\sum_k B_i^k + \mu(F)\right) + \delta + \gamma \quad &\leq \quad (B_i - \gamma) + \delta + \gamma \\ &\leq \quad B_i + \delta. \end{aligned}$$

Thus the black agent does not benefit from hiding any of its nodes. □

Theorem 2.2.1. *Mechanism BALANCE can be implemented in polynomial time, has an approximation ratio of 2, and is truthful.*

Proof. Let G be an arbitrary compatibility graph with n nodes. The truthfulness follows from Proposition 2.2.5. Note that BALANCE returns a maximal matching on all instances; and so for every edge that is matched in a maximum cardinality matching at least one node is matched in the output of BALANCE, thus the 2-approximation result follows.

We are left to argue that BALANCE can be implemented in polynomial time. At every step the mechanism increases the cardinality of the matching by 1 or then the algorithm stops. Given that a maximum cardinality matching has at most $n/2$ edges, then the number of augmentations is bounded. As finding an augmenting path can be done in polynomial time, BALANCE can be implemented in polynomial time. □

2.3 Mechanism BALANCE-AND-AUGMENT

Consider the following randomized mechanism, BALANCE-AND-AUGMENT, based on BALANCE:

1. Apply BALANCE on G to get the matching M.
2. Augment all augmenting paths in G with respect to M to get a maximum cardinality matching $\bar{M}$.
3. Select M with probability 1/2 and $\bar{M}$ with probability 1/2.

An illustrative example appears in Appendix A.

It is clear that this mechanism is not universally truthful, because it uses a max-cardinality matching with probability 1/2 and in all of those realizations the mechanism can in fact be manipulated by the result of Sönmez and Ünver [37].

Theorem 2.3.1. *The Mechanism* `BALANCE-AND-AUGMENT` *can be implemented in polynomial time, has an approximation ratio of 4/3, and is truthful in expectation.*

Proof. That `BALANCE-AND-AUGMENT` can be implemented in polynomial time is straightforward. Next we proceed to prove the approximation ratio of mechanism `BALANCE-AND-AUGMENT`. For any graph G, we let $\mu(G)$ be the number of nodes matched in a maximum cardinality matching of graph G. The expected number of nodes matched by `BALANCE-AND-AUGMENT` is at least $1/2\ \mu(G)$ with probability 1/2, and $\mu(G)$ with probability 1/2. Simplifying, we find that this mechanism matches at least $3/4\,\mu(G)$, proving the approximation ratio. Note that this mechanism always returns a maximal matching, and by a result of Caragiannis et al. [15], this is the best possible. The only property left to prove is that `BALANCE-AND-AUGMENT` is truthful in expectation.

Let M be the output of `BALANCE` on G and let $\bar{M}$ be the matching obtained by augmenting M to a max-cardinality matching. From Proposition 2.2.2, we know that `BALANCE` maximizes the gray score or the black score subject to matching all vital nodes in the graph; or return a maximum cardinality matching. We will consider each case separately. Suppose it maximizes the gray score while matching all vital nodes. We will argue that in this the case, the gray agent has no incentive to lie. From the definition of `BALANCE-AND-AUGMENT` in Step 1 and Step 2, all vital nodes of both agents are matched. Assume that gray agent has an incentive to lie. Denote the resulting matching as G'. This implies that there exist a matching of G that matches more gray nodes than M (equivalently $\bar{M}$) while matching all of the black agent vital nodes. Notice that black vital nodes in G are also vital nodes in $G' = \{G^k\}$. Let L be a matching that matches more gray nodes than M subject to matching all of the black agent vital nodes. Let us consider the symmetric difference between $\bar{M}$ and L. Given that $\bar{M}$ is a maximum cardinality matching of G it follows that the only components in which L matched more gray nodes than $\bar{M}$ is in the even length components in which one leaf is gray the other one is black and the gray leaf is matched by L. Moreover as L matches all black vital nodes, the black leaf node in such a component is not a vital black node. If M is not a maximum

cardinality matching this contradicts the definition of `BALANCE` because in Step 4 `BALANCE` could have alternated this component of the symmetric difference to match the gray leaf and augmented a black augmenting path (such a path exist because if M is not a maximum cardinality matching). If M is a maximum cardinality matching then M did not match the maximum number of gray nodes subject to matching all black vital nodes. We can conclude that the gray agent has no incentive to lie.

Then the only agent that could have incentive to lie is an agent that does not get the maximum possible score subject to matching all vital nodes. Let us assume without loss of generality that M does not match the maximum possible number of black nodes subject to matching all vital nodes. Assume that the black agent has an incentive to lie. We shall show that when the black agent hides a subset of nodes, the gain from this maneuver in Step 2 of `BALANCE-AND-AUGMENT` is offset by an equal or larger loss in Step 1. As the matchings computed in these steps are equally likely to be chosen, the result follows. Let H be the subgraph induced by the hidden black nodes (known only to the black agent). From Proposition 2.2.3 we can assume without loss of generality that black nodes that are matched by the black agent [4] were not reported to the centralized planner. Let M' be the output of `BALANCE` on G' together with the maximum cardinality matching among nodes hidden by the black agent and let $\bar{M}'$ be the matching obtained from augmenting all augmenting paths in M' together with the maximum cardinality matching among nodes hidden by black agent. Define $\Delta_{BR} := \min\{R_e - R_i, B_e - B_i, \frac{\mu(G)-B_i+R_i}{2}\}$. From the definition of `BALANCE`, it follows that $\mu_B(M) = B_i + \Delta_{BR}$: the last term accounts for the case when `BALANCE` finds a max-cardinality matching. Similarly, We define Δ'_{BR} as the number of gray nodes (black nodes) matched by `BALANCE` in G' besides the R'_i (B'_i) nodes that are matched in Step 1 of `BALANCE`.

We divide this proof into two cases. We first consider the case in which $\Delta_{BR} \geq \Delta'_{BR}$, then we consider the case in which $\Delta_{BR} < \Delta'_{BR}$, and argue that in both cases, black agent has no incentive to lie. Assume that $\Delta_{BR} = \Delta'_{BR} + \delta$ for some $\delta \geq 0$. We shall show that

$$\mu_B(\bar{M}') \quad \leq \quad \mu_B(\bar{M}) + \delta, \tag{2.3.1}$$

[4] Nodes matched by `BALANCE` and `BALANCE-AND-AUGMENT` are matched by a centralized planner

and that

$$\delta + \mu_B(M') \quad \leq \quad \mu_B(M). \tag{2.3.2}$$

Adding these inequalities, we find $\mu_B(\bar{M}') + \mu_B(M') \leq \mu_B(\bar{M}) + \mu_B(M)$, which implies the result. Inequality (2.3.2) follows trivially from definition of M and M':

$$\begin{aligned} \mu_B(M) - \mu_B(M') &= B_i + \Delta_{BR} - \sum_k B_i^k - \mu(F) - \Delta'_{BR} \\ &\geq B_i + \Delta'_{BR} + \delta - B_i - \Delta'_{BR} \\ &= \delta \end{aligned}$$

Now we are left to argue inequality (2.3.1). Observe that $\mu_R(M') = R_i + \Delta'_{BR} = R_i + \Delta_{BR} - \delta$; using $\mu_R(M) = R_i + \Delta_{BR}$, we see that $\mu_R(M) - \mu_R(M') = \delta$. But we already know that $\mu_R(\bar{M}) = \mu_R(M)$, and $\mu_R(\bar{M}') \geq \mu_R(M')$. Putting all this together, we have

$$\mu_R(\bar{M}) - \mu_R(\bar{M}') \ \leq \ \mu_R(M) - \mu_R(M') \ = \ \delta. \tag{2.3.3}$$

Note that $\mu_R(\bar{M}) + \mu_B(\bar{M})$ is the size of the max-cardinality matching in the original graph, and $\mu_R(\bar{M}') + \mu_B(\bar{M}')$ is the size of a matching of G when black agent matches some of his nodes internally, and the rest of the graph is matched optimally. Trivially, we have:

$$\mu_R(\bar{M}') + \mu_B(\bar{M}') \ \leq \ \mu_R(\bar{M}) + \mu_B(\bar{M}),$$

which when rearranged gives

$$\mu_B(\bar{M}') - \mu_B(\bar{M}) \ \leq \ \mu_R(\bar{M}) - \mu_R(\bar{M}') \ \leq \ \delta,$$

where the last inequality follows from (2.3.3).

Next we proceed to argue that if $\Delta'_{BR} = \Delta_{BR} + \delta$ for some $\delta \geq 0$, the black agent has no incentive to be untruthful. To do so we will prove that

$$\mu_B(\bar{M}') \quad \leq \quad \mu_B(\bar{M}) - \delta, \tag{2.3.4}$$

and that

$$\mu_B(M') \quad \leq \quad \mu_B(M). \tag{2.3.5}$$

Adding these inequalities and using the fact that $\delta \geq 0$, we find that $\mu_B(\bar{M}') + \mu_B(M') \leq \mu_B(\bar{M}) + \mu_B(M)$, which implies the result.

We claim that $\sum_k B_i^k + \mu(F) \leq B_i - \delta$ and that δ is the extra number of gray nodes matched in M' when compare to M. To see this, let us consider the symmetric difference between M and M'. If M is not a maximum cardinality matching, from assumption that M does not matches the maximum number of black nodes subject to matching all vital nodes, it follows that M matches the maximum number of gray nodes subject to matching all vital nodes. This latter fact together with the assumption that $\Delta'_{BR} = \Delta_{BR} + \delta$ for some $\delta \geq 0$ imply that M' matches exactly δ more gray nodes than M. Consider $M\Delta M'$, given that there is no augmenting path in G with respect to M with a gray leaf, it follows that there are at least δ even length paths with a gray leaf matched by M' and a black leaf matched by M; This node is a vital node for black in G. This implies that M' matches δ fewer black vital nodes than M and from Proposition 2.2.4 the desired result follows. If M is a maximum cardinality matching, then for M' to match δ additional gray nodes than M, it must match δ fewer black nodes than M. Given that from Step 2 to Step 4 algorithm `BALANCE` matched $\Delta'_{BR} = \Delta_{BR} + \delta$ black nodes, we can conclude that in Step 1 it matched at most $B_i - \delta$ black nodes.

Using this bound on $\sum_k B_i^k + \mu(F)$ together with the definition of M and M' it follows:

$$\begin{aligned} \mu_B(M') - \mu_B(M) &= \sum_k B_i^k + \mu(F) + \Delta'_{BR} - (B_i + \Delta_{BR}) \\ &\leq B_i - \delta + \Delta_{BR} + \delta - B_i - \Delta_{BR} \\ &= 0. \end{aligned}$$

We are left to show that Inequality (2.3.4) holds. From the relation between $\bar{M}'$ and M', it follows trivially that $\mu_R(\bar{M}') \geq \mu_R(M')$. From assumption that $\mu_R(M') = R_i + \Delta_{BR} + \delta$ and $\mu_R(M) = R_i + \Delta_{BR}$ it follows that $\mu_R(M') - \mu_R(M) = \delta$. Also, $\mu_R(M) = \mu_R(\bar{M})$. Putting these inequalities together it follows that:

$$\mu_R(\bar{M}) - \mu_R(\bar{M}') \;\leq\; \mu_R(M) - \mu_R(M') \;=\; -\delta.$$

Given that $\bar{M}$ is a maximum cardinality matching of G and M' is a matching of G, then it follows that:

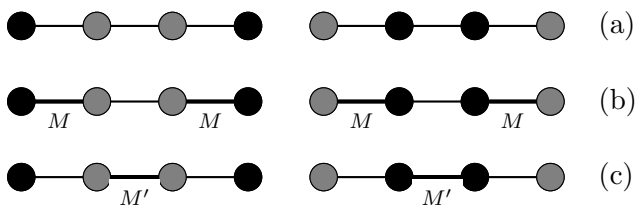


Figure 2.3.1: (a) Compatibility graph G which has two maximal connected components (b) Matching M is the output of mechanism `BALANCE` on G (c) Output of any strategy-proof mechanism that applies on each maximal connected component in isolation.

$$\mu_R(\bar{M}') + \mu_B(\bar{M}') \ \leq \ \mu_R(\bar{M}) + \mu_B(\bar{M}),$$

which when rearranged gives

$$\mu_B(\bar{M}') - \mu_B(\bar{M}) \leq \mu_R(\bar{M}) - \mu_R(\bar{M}') \ \leq \ -\delta.$$

This completes the proof of Proposition 2.3.1.

□

A nice and somehow surprising feature to be highlighted about the algorithm is that it does not require the graph to be connected: specifically if a graph has several maximal connected components, applying `BALANCE` or `BALANCE-AND-AUGMENT` on the entire graph may result in a different output than when applying these mechanisms on each maximal connected component of the graph as if they were in isolation. The relevance of this feature is that it could bring improvements in efficiency relative to mechanisms that work on each maximal connected component independently. This is illustrated in Figure 2.3.1.

2.4 Matchings on Paths and Cycles

The approximation ratio we establish on `BALANCE-AND-AUGMENT` is best possible, assuming the mechanism is required to find a maximal matching on all instances. While maximality is a natural requirement, it is interesting to ask if one can improve the approximation ratio if it is relaxed.

It is somewhat paradoxical that by matching "less" one can actually improve the worst-case performance, but this is simply the interplay between truthfulness and efficiency: by a threat of *not* matching an edge (even when such a match improves the performance on that instance), one can dissuade untruthful behavior in other instances, thereby improving the overall worst-case performance ratio. Motivated by this issue, we consider this question for path-cycle-graphs. Aside from their simplicity, one reason for studying these closely is that the worst-case examples so far have all been on path graphs. Our results show that indeed one can improve the worst-case performance if maximality is not imposed on the mechanism. This leaves open the possibility of a stronger performance guarantee on general graphs, but we have not been able to achieve that so far.
We first introduce a deterministic mechanism `SHARE` which will later be used by the our $\frac{5}{4}$ randomized mechanism `SHARE-BALANCE-AUGMENT`. Given a path-cycle graph P, we consider the following simple mechanism (mechanism `SHARE`) that always maintains a current matching M that gets updated in each iteration in the following way:

1. If there is a gray augmenting path with respect to M, augment M using that path. Repeat this step until the only augmenting paths left (if any) have at least one end-point black.
2. If there is a black augmenting path with respect to M, augment M using that path. Repeat these steps until the only augmenting paths left (if any) have at least one end-point gray.
3. If M has not been updated during this iteration, then STOP. Otherwise, go to Step 1.

In general graphs, mechanism `SHARE` is not truthful. To see how `SHARE` fails to be strategyproof, consider the graph G in figure 2.4.1. For general graphs the order in which the augmentations are done affects the score of the agents.

Notice that the approximation ratio of mechanism `SHARE` is infinite because when given a single edge between the gray and black agents, it does not match that edge! However, on path and cycles graphs, we can show that mechanism `SHARE` is truthful.

Proposition 2.4.1. *Mechanism `SHARE` is truthful on path-cycle graphs*

Proof. Let P be an arbitrary path and let M be the output of mechanism `SHARE` applied to P. We can assume that P is a connected graph, because applying `SHARE` to a non-connected graph is equivalent to applying `SHARE` to each one of its components independently. From the definition

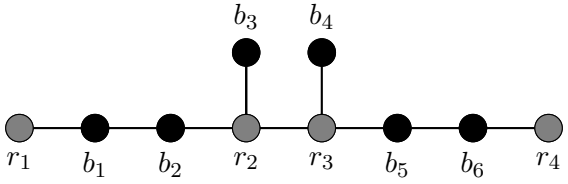


Figure 2.4.1: Compatibility graph G in which SHARE is not SP. SHARE fails in G because the matching that it returns depends on the order in which augmentations are done and so agent black has incentive to hide nodes b_1,b_2,b_5 and b_6 to match all its nodes with probability 1.

of SHARE it follows that step 1 (step 2) is equivalent to finding the maximum number of disjoint augmenting paths among nodes of agent gray (black) in P with respect to the matching M at the beginning of step 1 (step 2), breaking ties arbitrarily.

We first argue that the way in which ties are broken does not affect the score of the agents at termination. Consider iteration i of SHARE on P, and suppose the maximal set of vertex-disjoint augmenting paths respect to the matching M at the beginning of step 1 or step 2 is not unique. Step 1 and step 2 of SHARE at each iteration are symmetric, we can assume without loss of generality we are currently at step 2 of iteration i. The fact that the largest set of disjoint augmenting paths respect to the matching M at the start of step 2 of iteration i is not unique, implies that there exist at least one sub path of P, such that respect to matching M the following holds: 1) All its gray nodes are matched 2) It starts and finishes in an unmatched node of agent black 3) Each of its leaves is the leaf of at most one black-black augmenting path. Denote as $\left\{P'_j\right\}$ the set of sub paths of P such that 1), 2) and 3) are satisfied with respect to M at the start of step 2 of iteration i. From 1), 2) and 3) it follows that the sub paths in $\left\{P'_j\right\}$ are disjoint. From the definition of mechanism SHARE at most one node , a black one, will be unmatched in each sub path in $\left\{P'_j\right\}$ at termination of step 2 of iteration i. Let P^{i+1} be the path resulting from contracting in P each sub path in $\left\{P'_j\right\}$ in a node of agent black if a node in the corresponding element of $\left\{P'_j\right\}$ is left unmatched, and otherwise eliminating such a sub path from P. Clearly, the augmenting paths in P respect to M at the end of step 2 of iteration i are also in P^{i+1}, independently of which black node in each sub path in $\left\{P'_j\right\}$ is left unmatched. Thus at the end of iteration i, the same number

of nodes are matched independently of the way that ties were broken.

Now we will argue that not matching the maximum number of disjoint paths at step 1 or step 2 cannot increase the number of nodes matched for any of the agents with respect to matching M. Let us consider a modified version of mechanism `SHARE`, that we denote as `SHARE`', in which at each iteration the number of disjoint augmenting paths that are augmented at step 1 or step 2 is not necessarily the maximum number of disjoint augmenting paths that could be matched at that iteration. Denote as M' the output of mechanism `SHARE`' when applied on P.
We claim that the number of nodes of agent gray (black) matched in M' is not larger than the number of agent's gray (black) nodes matched in M. This claim will be proved by induction on the number of edges of P.

- In the *base case* we assume that P has one edge. In this case l and l' return a non empty matching only when the two nodes adjacent to the unique edge in P belong to the same agent. After the first iteration, M matches both nodes in P, implying the result.
- Suppose that for all paths or cycles with at most $k-1$ edges, M' matches at most the same number of nodes of agent gray (black) than M. Let P be an arbitrary path or cycle of length k and assume for the sake of contradiction that M' matches more nodes of agent gray than M.

  Let i be the first iteration of `SHARE`$'$ on P in which the number of gray nodes that could be matched at the end of step 1 exceeds the number of nodes matched in M. Denote as M'_i the matching at the beginning of iteration i during the execution of mechanism `SHARE`'. Let S be an arbitrary gray augmenting path in M'_i. We will now argue that either a gray augmenting path in M'_i could have been augmented during iteration $i-1$ or that the augmentation of such an augmenting path will not make the score of the gray agent at termination of iteration i larger than his score in M .

  - Suppose that S intersects a path that was found in a previous iteration, and it has not been augmented at the start of iteration i. These paths intersect in one leaf, so only one of these two paths could be augmented.
  - Suppose S does not intersect any path that was found in a previous iteration and it has

not been augmented at the start of iteration i. Denote as v_1 and v_2 the two leaves of S. If both leaves are not matched in M, it implies that in the sub path of P that starts in v_1 and finish in v_2, SHARE' matches more nodes of agent gray than mechanism SHARE, but this contradicts the induction hypothesis. This contradiction follows from the definition of mechanism SHARE; applying SHARE on P is equivalent to applying SHARE on the graphs $\{P\backslash v_1, \ldots, v_2\}$ and $\{v_1, \ldots, v_2\}$ independently. Up to iteration i applying SHARE' on P is also equivalent to applying SHARE' on graphs $\{P\backslash v_1, \ldots, v_2\}$ and $\{v_1, \ldots, v_2\}$ independently. Thus we can assume without loss of generality that v_2 is matched in M. From the definition of SHARE, v_2 was matched during an augmentation in which another node — denoted v_3 of agent gray was matched. We may assume without loss of generality that v_3 is matched in M'_i. Otherwise, it follows that SHARE' has matched up to iteration i at least two less black nodes than SHARE on the sub path $v_2, \ldots, v_3$, so augmenting S will make the score of the gray agent in M'_i at most the score of of the agent gray in M. Thus, v_3 is matched in M'_i and it was matched through an augmentation in which a node v_4 was augmented. Given that the graph is finite, eventually we will hit a node v_l that is matched in M'_i and not in M. Then v_1 and v_l are two nodes matched by M'_i and not by M in which all nodes in the sub path between nodes v_1 and v_l are matched and all of them have been matched through augmentations among nodes in the sub path of P between nodes v_1 and v_l. This implies that SHARE' matches more nodes in the sub path of P that starts in v_1 and finishes in v_l which contradicts the *induction hypothesis*. Then we can conclude that in this latter case augmenting path S cannot exist.

Now we are ready to argue that mechanism SHARE is SP. Suppose not, and that agent gray has incentive to hide nodes to increase its utility. Let P' be the subgraph of P induced by the nodes reported by agent gray and agent black. Given that all operations performed by SHARE on P' are valid operations in P, it follows that applying SHARE on P' is equivalent to applying SHARE on P but not necessarily augment all available augmentations at each step, which is the definition of SHARE'. Given that the number of nodes of any agent matched by SHARE' is at most the number of nodes of such agent matched by SHARE on P, then we can conclude that SHARE is SP. The same proof holds for cycles.

□

Despite its poor performance when used all by itself, we show that one can profitably use mechanism `SHARE` along with the mechanism `BALANCE-AND-AUGMENT`. Specifically, consider the following hybrid mechanism `SHARE-BALANCE-AUGMENT`. Let P be an arbitrary path-cycle graph.

1. Apply `BALANCE-AND-AUGMENT` on each maximal connected component of P to get the matching M_{12}.
2. Apply `SHARE` on each maximal connected component of P to get the matching M_3.
3. Select M_{12} with probability $4/5$ and M_3 with probability $1/5$.

Note that unlike mechanism `BALANCE` and `BALANCE-AND-AUGMENT`, mechanism `SHARE` applies to each connected component of a graph as if they were in isolation.

Proposition 2.4.2. *The Mechanism `SHARE-BALANCE-AUGMENT` can be implemented in polynomial time, has an approximation ratio of 5/4 and is truthful in expectation on path-cycle graphs.*

Proof. It is clear that `SHARE-BALANCE-AUGMENT` is truthful in expectation, as each of the component mechanisms satisfies that property and the probabilities are exogenously chosen. We show that mechanism `SHARE-BALANCE-AUGMENT` is in fact a 5/4-approximation.
Let G be an arbitrary maximal connected component of a path-cycle graph. Let M_1 be the output of mechanism `BALANCE` on G, let M_2 be the maximum cardinality matching resulted from augmenting all augmenting paths in G starting from matching M_1, and let M_3 be the output of mechanism `SHARE` on G. Let $N(M_1)$ be the number of nodes matched in matching M_1, similarly we define $N(M_2)$ and $N(M_3)$. To prove that `SHARE-BALANCE-AUGMENT` is a 5/4-approximation, is enough to show that $N(M_1) + \frac{1}{2} \cdot N(M_3) \geq N(M_2)$. This latter inequality together with the definition of mechanism `SHARE-BALANCE-AUGMENT` imply that the expected overall utility is lower bounded by $\frac{4}{5} N(M_2)$, as shown next:

$$\begin{aligned} \frac{4}{5}\left(\frac{1}{2}N(M_1) + \frac{1}{2}N(M_2)\right) + \frac{1}{5}N(M_3) &= \frac{4}{5}\left[\frac{1}{2}\left(N(M_1) + \frac{1}{2}N(M_3)\right) + \frac{1}{2}N(M_2)\right] \\ &\geq \frac{4}{5}N(M_2), \end{aligned}$$

which prove the approximation ratio of 5/4.

Claim 1.

$$N(M_1) + \frac{1}{2} \cdot N(M_3) \quad \geq \quad N(M_2) \tag{2.4.1}$$

Proof. Since G is a collection of paths or cycles any mechanism `BALANCE` terminates at the end of step 3 thus M_1 is a maximum cardinality matching or it matches all the nodes of one of the agents. In the former case the inequality (2.4.1) follows trivially, so we may assume that M_1 is not a maximum cardinality matching and that it matches all nodes of agent gray. To prove this claim let us consider the symmetric difference between M_3 and M_2.

- In an even length cycle or path component, as M_3 matches the same number of nodes as M_2, and M_1 is maximal, thus inequality (2.4.1) follows trivially.
- An odd length component of length *at least 3* starts and finishes with an edge of M_3: thus from definition of M_3, this component has a leaf that belongs to agent gray and a leaf that belongs to agent black. The leaf that belongs to agent gray is matched by M_1. From the maximality of M_1 and the fact that the graph is a path it follows that for every 3 consecutive nodes 2 nodes are matched by M_1. We will argue that inequality (2.4.1) holds in odd length components when the number of nodes matched in M_2 is at least 4.

$$\begin{aligned}
\frac{N(M_1) + \frac{1}{2} \cdot N(M_3)}{N(M_2)} &\geq \frac{\left\lceil \frac{2}{3} N(M_2) \right\rceil}{N(M_2)} + \frac{1}{2} \cdot \frac{N(M_2) - 2}{N(M_2)} \\
&= \frac{\left\lceil \frac{2}{3} N(M_2) \right\rceil}{N(M_2)} + \frac{1}{2} - \frac{1}{N(M_2)} \\
&= \frac{1}{N(M_2)} \left(\left\lceil \frac{2}{3} N(M_2) \right\rceil - 1 \right) + \frac{1}{2} \\
&\geq 1.
\end{aligned}$$

  For $N(M_2) = 4$, which arises when the component in consideration has 3 edges, the inequality holds tight. For $N(M_2) \geq 6$, which arises when the component in consideration has 5 edges or more, $\frac{1}{N(M_2)} \left(\left\lceil \frac{2}{3} N(M_2) \right\rceil - 1 \right) + \frac{1}{2} \geq \frac{7}{6} - \frac{1}{N(M_2)} \geq 1$.

- An odd length component of length 1 has only one edge that is incident to a node of agent gray and a node of agent black and belongs to matching M_2. From the relationship between M_2 and M_1 and the fact that M_1 matches all the gray nodes in the path, it follows that the gray node in this component is also matched by M_1. If this edge is in M_1, the claim follows

trivially. Otherwise there is an edge of M_1 that is adjacent to this component and also match the node of agent gray. Let us denote the edge that belong to M_2 as e_2 and the edge that belongs to M_1 as e_1. See Figure (2.4.2).

Let us consider the component of $M_2 \Delta M_3$ aside from e_2 that have a node in common with e_1. We will argue that this component together with edges e_1 and e_2 satisfies inequality (2.4.1).

- If there is no component of $M_2 \Delta M_3$ adjacent to $e_2 - e_1$, given that every node matched by M_1 is also matched by M_2 it follows that the edge of M_2 adjacent to M_1 is also in M_3. Let us denote the edge that is adjacent to e_1 and is not e_2 as (v_1, v_2), in which v_1 is the node also incident to e_1, see figure (2.4.2).

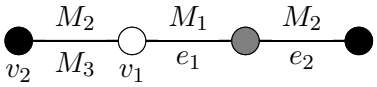


Figure 2.4.2: e_2 is matched by M_2, e_1 is matched by M_1 and (v_1, v_2) is matched by both M_2 and M_3. Additionally, v_2 is not matched by M_1.

We should assume that neither v_2 nor the black node incident to edge e_2 are matched by M_1. Otherwise it follows that in such component M_1 matches 3 nodes, M_3 matches 2 nodes implies that for this component $N(M_1) + \frac{1}{2} \cdot N(M_3) = 4$, which is equal to the number of nodes matched by M_2. If v_2 is not matched by M_1 then it follows that v_2 is a node of agent black, as otherwise there would be a black-gray augmenting path respect to M_1. If v_1 belongs to agent gray there are 2 consecutive nodes of agent R adjacent in both cases to a node of agent B and one of them is not matched, from definition of mechanism `SHARE`, these 2 nodes will be matched in the first iteration and will never be unmatched during the execution of the algorithm, so v_1 cannot be a gray node. As v_2 is a *vital* node and it is unmatched in the output of mechanism `BALANCE`, M_1, v_1 cannot belong to agent B. It follows that this cannot happen.

- If the component of $M_2 \Delta M_3$ adjacent to e_1 that does not contain edge e_2 has even length, let us denote the edge incident to e_1 that is not e_2 as (v_2, v_1) in which node v_1 is incident to edge e_1, see figure (2.4.3).

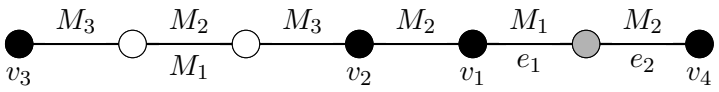


Figure 2.4.3: The component of $M_2 \Delta M_3$ that is adjacent to e_1 and does not contain edge e_2 has an even number of edges.

In this case we should assume that v_1 belongs to agent black, as otherwise there are 2 consecutive nodes of agent gray (nodes incident to edge e_1) adjacent in both cases to a black node, and none of them is matched in M_3 which contradicts mechanism `SHARE`, in which at least one node incident to e_1 would have been matched in the first iteration. Additionally, we can also assume that v_2 is a black node as otherwise there would be a gray-black augmenting path with respect to M_1, which is a contradiction. Finally we should also assume that the leaf of this component that is not incident to edge e_1, v_3, is also a black node: because it is not matched by M_2 and by M_1, it can not be a gray node because all gray nodes are matched by M_1.

From the maximality of M_1 it follows that for every 3 nodes in the component of $M_2 \Delta M_3$ adjacent to e_1 and that does not include e_2, at least 2 nodes are matched by M_1. Moreover, as v_3 is not matched by M_1 we can conclude that the length of the even component of $M_2 \Delta M_3$ adjacent to e_1 that does not include e_2 (subpath $v_3, \ldots, v_1$) is at least 4. This follows because if such component has length 2 then (v_1, v_2) is matched by M_1 which contradicts that e_1 is matched by M_1. The fact that length of such component is at least 4 and from the maximality of M_1, it follows that M_1 matches at least half the nodes matched by M_3 in the component of $M_2 \Delta M_3$ adjacent to e_1 that does not include e_2. Then it follows that:

$$\begin{aligned} N_1(v_3, \ldots, v_4) + \frac{1}{2} \cdot N_3(v_3, \ldots, v_4) &\geq \left(\frac{1}{2} \cdot N_3(v_3, \ldots, v_4) + 2\right) + \frac{1}{2} \cdot N_3(v_3, \ldots, v_4) \\ &= \left(\frac{1}{2} \cdot N_3(v_3, \ldots, v_1) + 2\right) + \frac{1}{2} \cdot N_3(v_3, \ldots, v_1) \\ &= N_3(v_3, \ldots, v_1) + 2 \\ &= N_2(v_3, \ldots, v_1) + 2 \\ &= N_2(v_3, \ldots, v_4), \end{aligned}$$

which is the desired result.

– If the component of $M_2 \Delta M_3$ adjacent to $e_2 - e_1$ has odd length of length at least 3, we claim that the second edge of the component of $M_2 \Delta M_3$ adjacent to e_1 that does not include e_2 is matched by M_1. Assume not. From the fact that v_1 is not matched by M_3

it follows that v_1 is a black node. If v_2 is gray there is a black-gray augmenting path in M_1. If v_2 is black, it must be vital and so it cannot be unmatched in M_1. Given that v_1 is also a node of agent black, it follows a contradiction with definition of mechanism `BALANCE` because a vital node, node v_2, is unmatched in M_1. See figure (2.4.4).

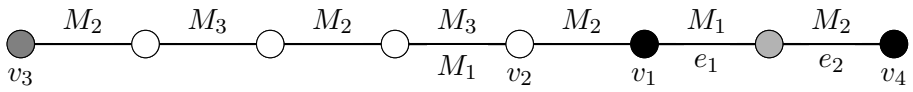


Figure 2.4.4: The component of $M_2 \Delta M_3$ that is adjacent to e_1 and does not contain edge e_2 has a odd number of edges.

Let us denote as v_3 the leaf of the component of $M_2 \Delta M_3$ adjacent to e_1 that is not e_2 that is matched by M_3. We claim that v_3 is matched by M_1 and that it is a gray node. If this is not the case, then v_3 it is a black node and given that v_1 is also a black node it follows that there is a black augmenting path with respect to M_3 which contradicts the definition of mechanism `SHARE`.

The number of nodes matched by M_1 among nodes $v_3, \ldots, v_2$ is at least:

$$N_1(v_3, \ldots, v_2) \geq 3 + \begin{cases} \frac{2}{3}\left(N_2(v_3, \ldots, v_1) - 4\right) & \text{if } N_2(v_3, \ldots, v_1) - 4 \bmod 3 \text{ is } 0 \\ \frac{2}{3}\left(N_2(v_3, \ldots, v_1) - 5\right) & \text{if } N_2(v_3, \ldots, v_1) - 4 \bmod 3 \text{ is } 1 \\ \frac{2}{3}\left(N_2(v_3, \ldots, v_1) - 6\right) + 2 & \text{if } N_2(v_3, \ldots, v_1) - 4 \bmod 3 \text{ is } 2 \end{cases}$$

From the fact that v_3 is matched by M_1 and the edge of M_3 that matches v_2 is also in M_1, it follows that in the worst case all nodes matched by M_1 in the sub path $\{v_3, \ldots, v_1\} \setminus \{v_3\} \cup \{e \in M_1 : v_2 \text{ is incident to } e\}$ are matched with another node in the sub path. There are $N(v_3, \ldots, v_1) - 4$ nodes in the component of $M_2 \Delta M_3$ in consideration that could be matched besides the three that we have argued are matched by M_1 in $v_3, \ldots, v_2$. From the maximality of M_1, at of every, 3 consecutive nodes, 2 are matched by M_1 and given that $N_2(v_3, \ldots, v_1) - 4$ is an even number, it follows that depending on the value of $N(v_3, \ldots, v_1) - 4 \bmod 3$ the three situations previously described will arise. In the first case, if $(N_2(v_3, \ldots, v_1) - 4) \bmod 3$ is 0, it follows that:

$$\begin{aligned} N_1(v_3, \ldots, v_4) + \frac{1}{2} N_3(v_3, \ldots, v_4) &= (N_1(v_3, \ldots, v_2) + 2) + \left(\frac{1}{2} N_2(v_3, \ldots, v_1) - 1\right) \\ &\geq \left(3 + \frac{2}{3}\left(N_2(v_3, \ldots, v_1) - 4\right)\right) + 2 + \frac{1}{2} N_2(v_3, \ldots, v_1) - 1 \\ &= N_2(v_3, \ldots, v_1) + 2 + \frac{1}{6} N_2(v_3, \ldots, v_1) - \frac{2}{3} \end{aligned}$$

$$\begin{aligned} &= N_2(v_3,\ldots,v_4) + \frac{1}{6}N_2(v_3,\ldots,v_1) - \frac{2}{3} \\ &\geq N_2(v_3,\ldots,v_4). \end{aligned}$$

The last inequality follows from the fact that $N_2(v_3,\ldots,v_4) \geq 4$ from assumption. In the second case, if $(N_2(v_3,\ldots,v_1) - 4) \bmod 3$ is 1, then it follows that:

$$\begin{aligned} N_1(v_3,\ldots,v_4) + \frac{1}{2}N_3(v_3,\ldots,v_4) &= (N_1(v_3,\ldots,v_2)+2) + \left(\frac{1}{2}N_2(v_3,\ldots,v_1) - 1\right) \\ &\geq \left(3 + \frac{2}{3}(N_2(v_3,\ldots,v_1) - 5)\right) + 2 + \frac{1}{2}N_2(v_3,\ldots,v_1) - 1 \\ &= N_2(v_3,\ldots,v_1) + 2 + \frac{1}{6}N_2(v_3,\ldots,v_1) - \frac{10}{3} \\ &= N_2(v_3,\ldots,v_4) + \frac{1}{6}N_2(v_3,\ldots,v_1) - \frac{4}{3} \\ &\geq N_2(v_3,\ldots,v_4). \end{aligned}$$

In which the last inequality follows because if $N_2(v_3,\ldots,v_1) - 4 \bmod 3$ is 1 then $N_2(v_3,\ldots,v_4) \geq 8$. Finally we consider the case in which $(N_2(v_3,\ldots,v_1) - 4) \bmod 3$ is 2. In this case it follows that:

$$\begin{aligned} N_1(v_3,\ldots,v_4) + \tfrac{1}{2}N_3(v_3,\ldots,v_4) &= (N_1(v_3,\ldots,v_2)+2) + \left(\tfrac{1}{2}N_2(v_3,\ldots,v_1) - 1\right) \\ &\geq +\tfrac{2}{3}(N_2(v_3,\ldots,v_1) - 6) + 2 + 2 + \\ &\quad \tfrac{1}{2}N_2(v_3,\ldots,v_1) - 1 \\ &= N_2(v_3,\ldots,v_1) + 2 + \tfrac{1}{6}N_2(v_3,\ldots,v_1) \\ &\geq N_2(v_3,\ldots,v_4). \end{aligned}$$

– If the component of $M_2 \Delta M_3$ adjacent to e_1, that is not e_2, has length 1, let us denote the nodes that are incident to this component as v_1 and v_2 in which v_1 is the node that is also incident to e_1. Given that neither v_1, v_2 nor nodes incident to e_1 and e_2 are matched by M_3, then in the subpath $v_2 v_1 - e_1 - e_2$ there are no two consecutive nodes of the same color because this will contradict the definition of mechanism `SHARE`. Also the leaf of this subpath that is not v_2 is a black node because it is not matched by M_1. Then it follows that v_2 is a gray node and so is also matched by M_1 with an edge $v_3 v_2$. Given that all nodes matched by M_1 are also matched by M_2 then v_3 is also matched by M_2. We repeat the argument just described until we encounter a node that is matched by M_3 and so we have encounter a component of $M_2 \Delta M_3$ with length at least 2, or an edge that is matched in both M_2 and M_3, or the entire graph is a path in which case there are no two consecutive nodes that belong to the same agent. See figure 2.4.5.

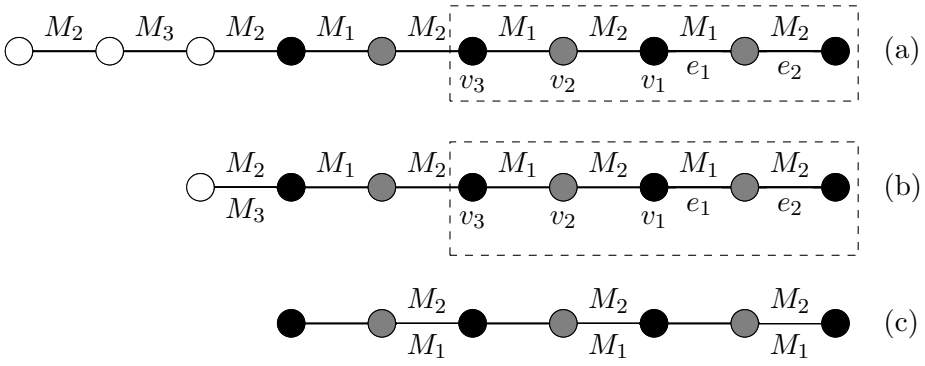


Figure 2.4.5: The component of $M_2 \Delta M_3$ adjacent to e_1 that is not e_2 has length 1. (a) the component of $M_1 \Delta M_2$ that includes e_2 intersect with a component $M_2 \Delta M_3$ with length at least 2 (b) the component of $M_1 \Delta M_2$ that includes e_2 intersect with an edge that is matched by both M_3 and M_2 (c) M_3 matches no node in the graph thus there is no two consecutive nodes in the graph that belong to the same agent, and M_1 and M_2 matches are equal.

If the graph is a path or a cycle in which there is no two consecutive nodes that belong to the same agent, then from definition of `SHARE`, it follows that M_3 matches no node in this graph and M_1 and M_2 are the same matching and so the desired results follows trivially.

We should consider the case in which there are at least two consecutive nodes in the graph that belong to the same agent, thus that M_3 is not empty. Let us denote as P_{BR} the subpath in $M_2 \Delta M_1$ that contains e_2 and has only one node that is also matched by M_3. Denote as Q_{BR} the subpath of P_{BR} of length three that has as leaf node x_1. The edge of M_2 incident to x_1 may be in a component of $M_2 \Delta M_3$ or may not, but at least one of its incident nodes is matched by M_3. In the former case, denote as R_{BR} such a component of $M_2 \Delta M_3$ and in the later case denote as R_{BR} this edge of M_2 that is also in M_3.

From definition of R_{BR} and Q_{BR} and the results previously proved in this claim in the case in which the component of $M_2 \Delta M_3$ adjacent to e_1 has length at least 2 it follows that:

$$[N_1(R_{BR} \cup Q_{BR}) - 1] + \frac{1}{2} \cdot N_3(R_{BR} \cup Q_{BR}) \quad \geq \quad N_2(R_{BR} \cup Q_{BR}). \tag{2.4.2}$$

From definition of P_{BR} it follows that $N_1(P_{BR} \backslash Q_{BR}) = N_2(P_{BR} \backslash Q_{BR}) - 1$. Adding

to the inequality (2.4.2) implies the desired result.

□

□

If we want a *universally strategyproof* mechanism on path-cycle graphs, we simply use the output of `BALANCE` with probability 2/3 and the output of `SHARE` with probability 1/3. Note that because `BALANCE` and `SHARE` are truthful mechanisms, this hybrid mechanism is in fact universally truthful. Its expected performance is:

$$\begin{aligned} \frac{2}{3}|M_1| + \frac{1}{3}|M_3| &= \frac{2}{3}\left(|M_1| + \frac{1}{2}|M_3|\right) \\ &\geq \frac{2}{3}|M_2|, \end{aligned}$$

in which the last inequality follows from Claim 1 in Proposition 2.4.2.

2.5 On strategy-proofness

In Section 2.1 we have defined the utility of an agent as the total number of its nodes that are matched by the mechanism, plus the nodes matched in a maximum cardinality matching of the graph induced by the nodes that were either not reported or left unmatched by the mechanism. In this definition of the utility of an agent, nodes that are not matched by the mechanism because they were not reported and nodes that were reported but left unmatched by the mechanism treated the same. However, due to the dynamic nature of the kidney exchange problem, an agent that internally matches a node that was already reported to the centralized planner will lose trust from the centralized planner once this becomes known.

We incorporate this consideration into the model by forbidding the agent from using the reported nodes left unmatched by the mechanism. In other words, the utility of an agent is the total number of its nodes that are matched by the mechanism, plus the nodes matched in a maximum cardinality matching of the graph induced by its nodes that were not reported. It is clear that the definition of utility of an agent has direct impact in the required conditions for strategy-proofness of a mechanism and so we need to distinguish strategy-proofness in the 2 situations. In the former case we will say

that a mechanism is *strategy-proof in the strong sense* and in the latter case a mechanism is *strategy-proof in the weak sense*. Similarly we extend these notions to strategy-proofness in expectation.
It ts also clear that mechanisms that are *strategy-proof in the strong sense* are also *strategy-proof in the weak sense*, so what we would like to analyze is the improvement in efficiency that can be achieved with a weaker definition of strategy-proofness.

2.5.1 BALANCE-ALL and BALANCE-ALL-AND-AUGMENT

We argue that a modified version of mechanism BALANCE provides strategy-proofness in the weak sense, even though for some instances it provides an incentive to the agents to report a node that is finally left unmatched. Intuitively someone could see such a node as a warranty for the mechanism to provide a higher utility for both agents and still preserve truthfulness.
We define mechanism BALANCE-ALL as a modified version of mechanism BALANCE in which V_B is defined as the set of black nodes reachable from some node in U_R via an even-length alternating path with respect to the matching at start of step 4. From the definition of BALANCE-ALL it follows that it matches at least the same number of nodes for each agent that BALANCE and thus lower bounds on the approximation ratio using the stronger definition of utility also apply. To see how BALANCE can create incentive for an agent to report a node that is left unmatched and how BALANCE fails to be strategy-proof in the strong sense see Figure 2.5.1.

As a natural extension, we define mechanism BALANCE-ALL-AND-AUGMENT as a modified version of mechanism BALANCE-AND-AUGMENT in which mechanism BALANCE is substituted by BALANCE-ALL. We proceed to present our results under the weaker notion of truthfulness. Due to the close similarity between the strategy-proofness proof of BALANCE and the strategy-proofness proof of BALANCE-ALL in this section we will just state our results and in Appendix B we present the proofs.

Proposition 2.5.1. *Let $f(G)$ be the output of the algorithm BALANCE-ALL on the graph G. Then either $f(G)$ is a maximum-cardinality matching, or it maximizes the number of matched nodes for at least one of the agents. In other words, $f(G)$ maximizes the total score, or the gray score or the black score.*

Proposition 2.5.2. *Mechanism BALANCE-ALL-AND-AUGMENT is strategy-proof in the weaker sense and is a 4/3 approximation mechanism.*

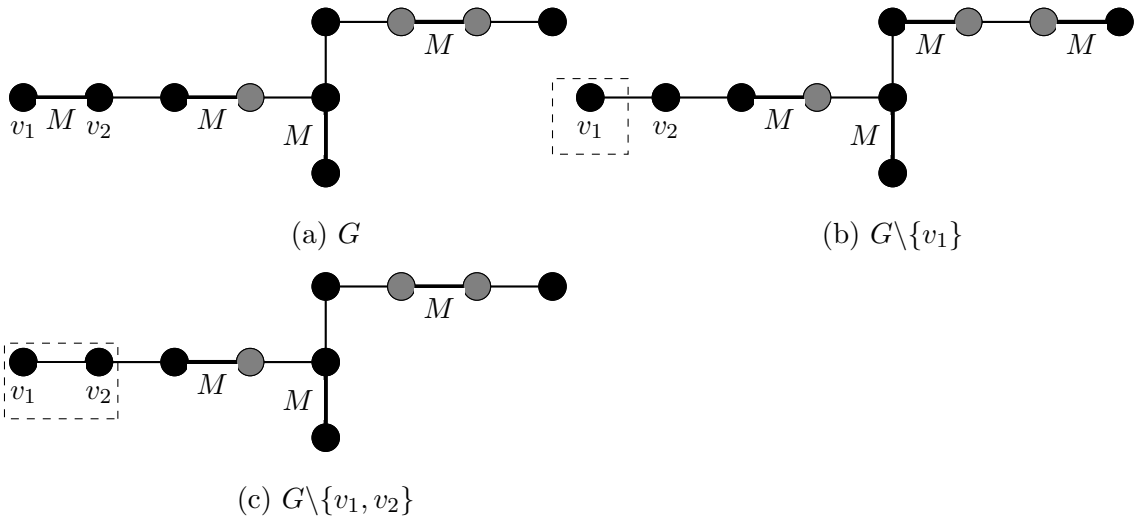
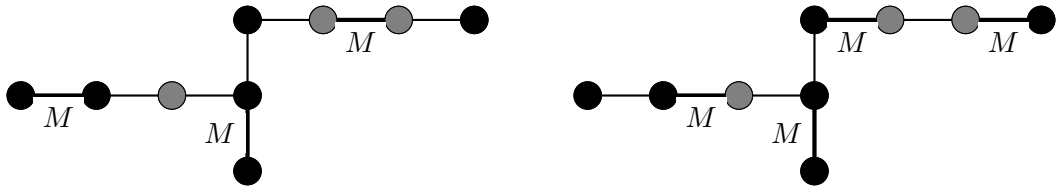


Figure 2.5.1: (a) Output of mechanism `BALANCE` on compatibility graph G, utility of agent black is 5 (b) Output of mechanism `BALANCE-ALL` on $G\backslash\{v_1\}$. The utility of agent black is 7, 5 nodes matched by `BALANCE-ALL` and 2 nodes matched by the agent black, the unmatched node v_2 matched with the non reported node v_1 (c) Output of mechanism `BALANCE-ALL` on $G\backslash\{v_1, v_2\}$. The utility of agent black is 5, 3 nodes matched by `BALANCE-ALL` and 2 nodes matched by agent black, the non reported nodes v_1 and v_2, thus agent black under mechanism `BALANCE-ALL` has incentive to report node v_2 which could remain unmatch to match it with non reported node v_1.



(a) Output of BALANCE on G (b) Output of BALANCE-ALL on G

Figure 2.5.2: (a) Output of mechanism BALANCE on compatibility graph G, utility of agent black is 4 (b) Output of mechanism BALANCE-ALL on compatibility graph G. The utility of agent black is 5.

Proposition 2.5.3. *Mechanism SHARE-BALANCE-ALL-AUGMENT is strategy-proof in expectation in the weaker sense and is a 5/4 approximation mechanism.*

Note that there is no need to make a similar extension to SHARE-BALANCE-AUGMENT because given that it only applies to paths or cycles for which mechanism BALANCE and BALANCE-ALL are the same. This equivalency follows from the fact that for paths and cycles no node is matched during execution of step 4. As a consequence, the proof of Proposition 2.5.3, in which we prove that mechanism SHARE-BALANCE-ALL-AUGMENT is a 5/4 approximation (strategy-proofness in expectation follows as a direct consequence of strategy-proofness in expectation of BALANCE-ALL-AND-AUGMENT and SHARE), mimics closely the proof of Proposition 2.4.2 and we omit it.

Notice also that even though the weaker notion of strategy-proofness improves the overall utility for some instances (see Figure 2.5.2 for an example) it does not affect the worst case performance.

2.6 Discussion

While we focused only on the case of two agents, intriguing questions remain, even for this special case. Is it possible to get a 5/4-approximation algorithm that is truthful in expectation for general graphs? We showed that this is possible for paths and cycles, but it is not clear how to address the general case. If we strengthen the truthfulness requirement to universal truthfulness, can we get a 3/2-approximation for general graphs? Again, we could do this for paths and cycles, but not in general. A problem of substantive interest is to better understand the case of more than 2

agents. Specifically, is there an improved randomized algorithm for an arbitrary number of agents? The best known bound is a 2-approximation. Finally, can one show non-trivial lower bounds on approximation ratios for more than two agents? Or on other graph topologies?

Chapter 3

A Dynamic Framework for the Design and Evaluation of Kidney Exchange Programs

3.1 Introduction

In most of the literature on the kidney exchange problem, strategies have been evaluated by their performance on a randomly generated large compatibility graphs; conclusions and recommendations have been drawn based on such a model. In reality, pairs arrive and depart over time, matching algorithms are run at different points in time, and the compatibility graph just before running a matching not only depends on the pairs that have arrived since the last run, but also on the pairs that were not matched in the previous runs, moreover the actual rate of arrival is a lot lower than the projected ones. Different strategies may benefit/hurt different type of pairs over time, and those outcomes typify the size and the blood type-sensitivity level composition of the compatibility graph over time. Data-based dynamic evaluations have been conducted by Dickerson et al. [10; 16; 18; 17]. Ünver [39] considered a dynamic model in which pairs arrive following a Poisson process. He designed an optimal dynamic policy under some assumptions such as the impossibility of tissue type incompatibilities, no departures and that there are arbitrarily many under demanded pairs in the exchange pool in the long run. He also shows that in his model it may not be always an

optimal strategy to match as many pairs as possible at a given point in time.

Our goal is to design a dynamic framework for the evaluation of extensively used kidney exchanges strategies through performance measures across the different type of pairs, while modeling features like departures, tissue type incompatibilities, and data-based rates of arrivals. We would like to provide a better understanding of the effects of these different features on the long-run performance of policies, and ultimately provide recommendations for policy makers and healthcare system administrators based on such a model.

We start by defining in Section 3.2 the model and the methodology for estimating the key parameters of the model. Our simulation experiments and findings are described in Section 3.3 and 3.4, respectively. In Section 3.5 we discuss our findings.

3.2 The model

As the kidney exchange market is still in a very early stage of development, it is difficult to understand the long-run effects of the strategies that are currently used given the small amount of data. However, if we assume that the characteristics of pairs that have participated in the market until now, are representative of the participants (patient-donor pairs, altruistic donors) that will join in the future, then we could make inferences about the evolution of the exchange market under different scenarios.

In this section we first describe the model elements: the patient-donor pairs and the underlying compatibility graph. We also discuss the model's fixed and variable parameters and describe how we estimate their values and range of values respectively.

3.2.1 Description

The pairs In our model, each pair is characterized by its *patient-donor blood type* and by its *patient's cPRA* (Calculated Panel Reactive Antibody[1]), which ranges from 0 to 1 and accounts for the patient's probability of rejecting immunologically, a blood type compatible potential donor.

[1] " The Calculated Panel Reactive Antibody (cPRA) is used in the allocation of kidney and pancreas organs as a measure of sensitization level. The CPRA estimates the percentage of donors that would be incompatible with the candidate, based on the candidate's unacceptable antigens."

We model a pair's patient cPRA as a discrete random variable with 3 possible values; each pair is classified as a high sensitized, medium or low sensitized, and each sensitization level has associated PRA value. Even though this classification is simplistic, it allows us to better understand the effect of the policies on pair types due exclusively to its sensitization level.
Pairs are classified in 48 different types according to their donor's and patient's ABO classification as well as their cPRA value. We assume that pairs arrive to an exchange pool in a Poisson manner and are independently classified as one of the 48 PRA-ABO types according to a PRA-ABO inclusion probability estimated from publicly available data. To estimate the pairs's arrival rate, we use information reported by the National Kidney Registry(NKR) [2] about their arrival rates and about their relative size, compared to other exchange programs in the country.

The PRA-ABO inclusion probabilities are a function of the patient's ABO blood type probability distribution, the donor's ABO blood type probability distribution and the patient's sensitization level probability distribution. We suppose that these probability distributions are independent. We estimate the patients' and donors' ABO blood-type distribution from the United Network for Organ Sharing (UNOS)[3] waiting list patient's ABO blood types distribution and living donors blood type distribution, respectively. The patient's sensitization level distribution is modeled as a variable to incorporate the ongoing discussion about its criticality in the sufficiency of small exchanges to clear the pool; however the corresponding PRA values are estimated from information reported by the main kidney exchange programs, and are considered constant across all the studied scenarios. We suppose that these probability distributions are independent. The methodology to model and estimate features and parameters of altruistic donors is analogous to the one used for pairs.

The compatibility graph The compatibility among pairs is modeled by a directed graph (compatibility graph) in which each node represents a patient-donor pair and there is an edge from a pair u to a pair v if the donor of pair u is blood type and tissue type compatible with the patient of pair v. Once a pair arrives to the exchange pool, we execute a Bernoulli trial with each one of the pairs whose donor is blood type compatible with the pair's patient to determine incoming arcs to the node associated with the newly arrived pair. We also perform a Bernoulli trial with each

[2] http://www.kidneyregistry.org/

[3] http://www.unos.org/

one of the pairs whose patient is blood type compatible with the donor of the newly arrival pair to determine outgoing arcs from the node associated with the newly arrived pair.

There are many other factors beyond blood type and tissue type compatibility involved in the completion of a transplant. We model these factors through a post-matching failure probability. Specifically once a matching policy is run on the compatibility graph at an instant in time, the post-matching failure probability accounts for the probability that at edge is not available at the moment of the transplant. Among the reasons for a post-matching failure to occur are the decline in health condition of any of the involved agents, changes in tissue compatibility, reneging by a pair's donor, or simply because the pair declines the matching offered kidney.

3.2.2 Parameters estimation

Arriving patient's ABO distribution We forecast the number of yearly arrivals of each ABO type based on a model fitted to yearly past observations of the additions by ABO blood type to UNOS's waiting list from the year 1995 to 2012. We only consider additions that corresponds to patients joining the waiting list for the first time. With this decision we aim to wane the effect of specific policies in our characterization of the stream of arriving patients. Intuitively, the component of the stream of arrivals composed by patients that were already served by the the UNOS's waiting list is not independent of the UNOS's assignment policy. Notice that the incentive for a patient to join the UNOS's wait list for the first time may not be independent of the assignment policy. A patient may feel that under the current policy they may be treated unfairly, and decide to not join the waiting list. However the fact that joining the waiting list has no cost, and that an offer can be rejected, supports the hypothesis that the effect of the current assignment policy on the decision of whether or not to join the wait list is minimal. From the predicted number of arrivals of each ABO blood type for our study time horizon, we determine the proportion of each ABO blood type of patients that is expected to arrive. Appendix C has a detailed description of our methodology.

Arriving donor's ABO distribution We based the estimation of the willing donor's ABO blood type distribution on the proportion of each ABO blood type in the yearly expected stream of living donor's arrivals to the UNO's program in our study time horizon. To do so, we forecast the number of yearly living donor arrivals to UNO's program based on a model fitted to yearly past

ABO	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
O	**49.07**	49.03	49.10	49.09	49.11	49.11	49.11	49.12	49.13	49.14
A	**32.44**	32.43	32.43	32.43	32.43	32.42	32.42	32.41	32.41	32.41
B	**14.66**	14.72	14.66	14.76	14.66	14.67	14.67	14.67	14.67	14.67
AB	**3.83**	3.82	3.81	3.81	3.80	3.8	3.8	3.8	3.79	3.78

Table 3.1: Forecasted percentage of Patient's arrivals of each ABO blood type in the next 10 years.

observations of the living donor's by ABO blood type that were transplanted in UNOS's program from 1995 to 2012. From the predicted number of yearly arrival by blood type we compute the proportion of those out of the total stream of arrival that are of each blood type in year 2013, and use this as the estimator of the ABO distribution of willing donors in the exchange pool. Notice that we estimate the ABO blood type distribution of the living donors that are suitable to go through a transplant from the data of living donors that actually went through a transplant.

As just mentioned, a downside of estimating the distribution of blood types among living donors from this data, is that the ongoing policies of UNOS's and its associated transplantation programs have an impact on the ABO blood type distribution of those living donors that get transplanted. Specifically, the requirements for a living donor candidate to become a viable donor depend on the guidelines of a program. Although this is a relevant remark, it is out of the scope of our study to evaluate when a living donor candidate actually becomes a donor. Precisely, for this work we consider only those living donors candidates that became viable donors according to the transplant program's guidelines.

Another potential flaw of the use of this data to estimate the blood type distribution of viable living donors is the fact that this estimate does not consider living donors who are willing to donate, but for various reasons (reneging, sickness, etc) they never get transplanted. The NKR reported in [29] that among 217 patient-donor pairs that were transplanted through chains, only 7 bridge donors became unavailable. Additionally, unlike the exchange programs which need to not only find a suitable donor for a patient in a patient-donor pair but also find a suitable patient for the willing donor, there is no such constraint in the UNOS's waiting list. This fact coupled with the fact that there are many patients of each blood type waiting to be transplanted in the UNOS's

waiting list, suggest that unless the living donor makes a late withdrawal, we can assume that all suitable living donors will go through transplantation.

Based on these facts and information reported by the kidney exchange community experience, we will assume that the living donors that goes through a transplant in the UNO's waiting list are representative of the population of living donors that are suitable to go through a transplant (a transplant that not necessarily occurs).

From the predicted number of living donors by ABO blood type that will be transplanted in the 2013-2022 time horizon, we estimate the yearly proportion of living donors of each ABO blood that are expected to be transplanted in our study time horizon. For our study we assume that

ABO	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
O	**65.52**	65.55	65.58	65.60	65.63	65.63	65.65	65.67	65.69	65.68
A	**26.26**	26.28	26.29	26.30	26.31	26.31	26.32	26.32	26.32	26.32
B	**7.47**	7.45	7.44	7.43	7.41	7.39	7.38	7.37	7.36	7.36
AB	**0.75**	0.72	0.70	0.67	0.65	0.65	0.65	0.64	0.63	0.64

Table 3.2: Forecasted percentage of Donor's arrivals of each ABO blood type in 10 years time horizon

across all the considered scenarios the ABO composition of the stream of arrivals of willing donors is determined by the percentage of the living donors that are forecasted to join the UNOS's waiting list in 2013, see Table 3.2.

Departures Pairs depart from the exchange pool without a transplant offer because of the patient or the donor's medical condition, death or their own will. In our model we just consider departures without a transplant offer due to the patient's medical condition, specifically when a patient is too ill or when he or she dies. We do not model departures without a transplant offer due to donor related reasons mainly because of the ethical issues related to donor's coercion or change in medical condition. Specifically, a policy should not be penalized for not making an earlier offer to a pair with a donor that reneges or got sick while waiting.

A pair departs without a transplant offer because of the patient's medical condition mainly when

the patient is too ill to receive a transplant, or when the patient dies. The patient's rate of departure because of the death is estimated from the UNOS's waiting list. This result is a good estimator of the rate at which pairs leave the exchange pool as consequence of the patient's death. The mortality rate attending for a transplant can be considered independent of the transplantation system (exchange pool or waiting list). Particularly this holds under the assumption that patients waiting to be transplanted are on similar transient medical treatments (dialysis), whether they are waiting in the exchange pool or in the waiting list. The patient's annual death rate in the UNOS's waiting list is reported in [19], see Table 3.3. According to UNOS's "the death rate is calculated by dividing the number of patients who died in a given year by the sum of the years that patients spend waiting and then multiplying by 1,000". Notice that this is a rate per 1,000 patients. More details about the methodology to compute the patient's death rate in the UNOS waiting list can be found in [19]. Innovation and new medical findings have made important contributions to this area,

year	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008
rate	84	73.6	73.5	76.8	75	73.4	70.5	70.4	65.0	63.3

Table 3.3: Number of deaths for every 1,000 patient-years on the waiting list, 1999 to 2008

which can be observed in the negative trend of the rate of deaths on the waiting list in 1999-2008 period. In this work we set the rate of departure due to the at *63.3 deaths/year* (the annual death rate per 1,000 patient-years at risk in 2008 report by UNOS) .

To model the departures from the exchange pool because the pair's patient is too ill to get a transplant, we define a *threshold waiting time* (taken to be 3 years), after which a patient is no longer suitable for a transplant. We justify this as follows: 22% of the patients in UNOS's active waiting list at the end of 2008 have waited more than 3 years, and 6.9% of the patients have waited more than 5 years. Patients that fell in this category have been on dyalisis for at least 3 to 5 years and so their health conditions and long-term survival expectation have been strongly undermined, see [28] and [41]. Thus, we can conclude that patients who have waited for so long are at high risk of departing the system without being served, but moreover that they were healthy enough when they joined the waiting list to be able to wait for such time as active patients. This suggests that

the policy failed these patients.

Additionally, for patients with a willing donor, exchanges pools are seen as an "improved" alternative to the UNOS waiting list. Thus, we expect any reasonable policy for the exchange pool to reduce the waiting time with respect to the expected waiting times on the UNOS wait list (which is independent of the fact that a patient has or not a willing donor). Likewise The threshold waiting time also serve as benchmark for policy evaluation.

Donors cPRA distribution As said earlier in section 3.2.1, pairs are characterized by their patient-donor blood type and by their cPRA. Next, we describe how our model determines that the donor of another pair is a suitable donor for a recipient of a pair given that they are blood compatible, by making use of the information provided by the cPRA of the patient. We use the cPRA of a patient as an estimator of the likelihood that the patient and a blood compatible donor are tissue type incompatible. The cPRA is defined by the HRSA/OPTN as the "likelihood that the recipient and donor would be incompatible". Notice that the cPRA already accounts for blood type incompatibility. This latter fact implies that while using the cPRA as an estimator of the likelihood that a patient and a blood compatible donor are tissue type incompatible, we are over estimating the probability of a recipient and a blood compatible donor to be incompatible. This result in a more pessimistic model.

To estimate the cPRA's distribution, we use as reference the data reported by the NKR in [29] that includes the cPRA values of the registered pairs in their program at the time of the 100th transplant, classifed in 3 categories: 0-9%, 10-79% and 80%-100%. Even though the sample has only 172 pairs, the results are consistent with the PRA data reported in [19] under the same classification. We choose as representative value of each cPRA category its third quartile, under the assumption that cPRAs are uniformly distributed in each category, see Table 3.4.

cPRA value	Probability
7.5%	0.55
62.5%	0.17
95%	0.28

Table 3.4: cPRA probability mass function used in our model

Pairs' rate of arrivals The NKR reported the number of new pair's registrations per month from January 2010 to June 2013 in [30]. For this work we use the average number of pairs that arrived to the NKR exchange program from January 2013 to June 2013 as the *pair's arrival rate*. This period of time presents the largest average over any contiguous 6 months period of time in the reported data. The data reveals a trend of growth over time of the number of registrations in the NKR, see Table 3.5.

Jan 13	Feb 13	Mar 13	Apr 13	May 13	Jun 13	Average
26	50	45	44	29	34	38

Table 3.5: New pair registrations per month in the NKR exchange program from January 2013 to Jun 2013

Post matching failure probability According to [30], in 2012 the NKR made 1623 individual matches offers out of which only 226 end up in a transplant. Among these 1623 offers, 692 were requested a Lymphocite Crossmatch test (XMs). Notice that offers are made based on a computer software which consider a large amount of information of the patient's history. As the immunological characterization of a patient changes over time, some patients are requested an additional test called XM. It is also reported in [30] that an unacceptable XM is the largest primary cause of swap failures in the NKR. We model this feature through a *post matching failure probability*.

3.2.3 Altruistic Donors

The number of altruistic donors has shown an increasing trend over the years, and there is an ongoing debate about how they should be allocated. Specifically, it is not clear if an altruistic donor's kidney should be allocated to someone in the waiting list or if it should be allocated to a pair in the exchange pool to initiate a chain of exchanges, which could end up with a donation to the waiting list. It is not clear how much we gain in efficiency and what the improvement in waiting times are for pairs in the exchange pool due to the participation of altruistic donors. On one hand, when an altruistic donor is used to launch a chain of exchanges, it removes the synchronous transplants constraint. On the other hand, the decision of using an altruistic donor to catalyze

a chain instead of donating directly to the waiting list, *may* hurt patients in the waiting list. To illustrate this latter point, consider the case in which an O type altruistic donor is used to launch a chain that ends with a non-O donation to the waiting list. This may benefit patients in the wait list that will move forward in the queue because pairs in front of them received a transplant through a chain (patients who were listed in the wait list and also in an exchange program, and who participated in the chain). However, this will hurt patients, particularly O patients, without an intended donor that have higher priority than any patient who participated in the chain. The overall effect in welfare is not clear. With the inclusion of a parameter to control the percentage of arrivals that corresponds to an altruistic donor we intend to measure the long-run gain in efficiency and waiting time improvement due to the allowance of altruistic donors initiating chains in the exchange pool. Considering that protective policies should not only guarantee overall improvement in overall measures but also guarantee acceptable performance measures to vulnerable patients, we analyze efficiency and waiting times across the 64 ABO-PRA patient-donor pairs in our model. To do so, we study the performance of various policies with and without altruistic donors. In this work we assume that 1% of the stream of arrivals are altruistic donors with ABO blood type distribution in Table 3.7. This analysis allows us to asses the relative benefits of diverting altruistic donors from the wailing list. Also, we asseses the dependence of those "benefits" on market parameters such as the rate of arrivals, the frequency at which matchings are decided, and whether or not compatible pairs participate in the exchange market.

3.2.4 Compatible pairs

Patient-donor pairs that are compatible do not need to join an exchange program. Although an exchange program may increase the chances of improving the quality of the match and benefit another recipient who is incompatible with his or her intended donor, it also burdens the process: longer waiting times, more involvement of the donor (more blood draws, days off work etc.) and donor's coercion. In this work we study the dynamics of the exchange pool in both cases: when compatible pairs join the pool, and when they do not. We define the random variables X_{p} and X_{PRA} to be the blood type and cPRA level respectively of an arriving pair's patient, and the random variable X_{d} to be its donor's blood type. From the model definition $X_{\mathrm{p}}, X_{\mathrm{d}}$ takes values in $\{O, A, B, AB\}$ and X_{PRA} is high, medium, or low as in Table 3.4.

Compatible pairs join the pool To model this, we assume that the blood type of an arriving patient is independent of the blood type of its intended donor. Thus,

$$P\left(X_{\mathrm{p}}=x, X_{\mathrm{d}}=y, X_{\mathrm{PRA}}=z\right) \quad = \quad P\left(X_{\mathrm{p}}=x\right) P\left(X_{\mathrm{d}}=y\right) P\left(X_{\mathrm{PRA}}=z\right),$$

for all $x, y \in \{O, A, B, AB\}, z \in \{H, M, L\}$. At first sight, this scenario does not seem to be of interest if the ABO distribution of patients and donors are estimated to be identical, because in the long run the expected number of pairs with patient ABO 'x' and donor ABO 'y' that have arrived is equal to the expected number of pairs with patient ABO 'y' and donor ABO 'x'. Thus, matching symmetric pairs should be enough to clear the exchange pool. This conclusion, however, does not tell the whole story. (1) Pairs arrive and depart over time; (2) Long waiting times on dialysis hurt the chances of a successful transplant; and (3) The arrival rate of different pair types is significantly different. The dynamics have an important effect on who waits and for how long etc. For instance, the average rate of arrivals to NKR from January 2013 to June 2013 is 38 patients per month, see [30]. Using as estimate of the distribution of donors' ABO, the ABO distribution of living donors to UNOS waiting list from 1998-2009, and as estimator of the distribution of patients ABO the ABO distribution of patients in UNOS waiting list during the same period of time. For example, the expected inter-arrival time of a pair with patient type B and donor type AB in our model is 8.8 months, whereas the expected inter-arrival time of its ABO symmetric pair is 9.4 months. This together with the fact that ABO compatibility is not enough for pairs to be compatible and that waiting time is a key factor in the graft survival rate suggest that the scarcity of pairs' types over time plays an important role in the design of policies and long-run behavior of the exchange pools. Also, the dynamic nature of the problem introduce complications that cannot be visualized in a static model, or in a dynamic model that does not model departures and patients' sensitization levels.

Compatible pairs don't join the pool For the scenarios in which compatible pairs don't join the pool, we assume that the blood type distribution of the general population of willing donors as well as the patients is in Table 3.7. However, the distribution of the willing donors that *actually* join the exchange program is dependent on the blood type distribution of the patients as well as the cPRA distribution. We define random variable D_g to be the blood type of a living donor randomly selected from the general living donor population. For an arbitrary $x \in \{O, A, B, AB\}$ we define

$S(x)$ as the maximal subset of $\{O, A, B, AB\}$ such that a donor with blood type in such subset is blood type compatible with a patient whose blood type is x. We model the dependency of X_{d} on X_{p} and X_{PRA} by conditioning the distribution of the willing donors that enter the exchange on the blood type of the patient:

$$P\left(X_{\mathrm{d}} = y | X_{\mathrm{p}} = x, X_{\mathrm{PRA}} = z\right) \quad = \quad \begin{cases} \frac{P(D_{\mathrm{g}}=y)}{1-\sum_{w\in S(x)} P(D_{\mathrm{g}}=w)\,(1-z)} & \text{If } y \notin S(x) \\ \frac{P(D_{\mathrm{g}}=y)\,z}{1-\sum_{w\in S(x)} P(D_{\mathrm{g}}=w)\,(1-z)} & \text{If } y \in S(x) \end{cases}$$

Recall that the distribution of D_{g}, X_{p} and X_{PRA} are given in Tables 3.4 and 3.7.

3.2.5 Matching policies

We consider three categories of matching policies: 1) Pairs are matched using cycles of arbitrary length, 2) Pairs are only matched through cycles of length at most 3 or 5, and 3) Pairs are matched through cycles of length at most 3 or 5, and through chains. Notice that chains are possible only in the presence of altruistic donors. As discussed in Section 2.1, the logistics involved in the execution of the exchanges in a cycle are one of the main challenges faced by exchange programs. We use as benchmark, in efficiency as well as in waiting times assessments, the results obtained using a matching policy in which cycles and chains of any length can be used at any point in time. Even though all the matching policies considered in this work match myopically - they decide matchings considering only pairs currently in the compatibility graph, and does not use available information about potential future arrivals - we capture the non-myopic effect when running matchings at different frequencies. In [18] Dickerson, Procaccia and Sandholm provide evidence of gain in efficiency due to the use of non-myopic matching policies; however, exchanges mostly match myopically in practice.

Next we proceed to describe our matching policies. Before doing so, we introduce some useful definitions. Let $G(t) = (N(t), E(t))$ be the compatibility graph at time t. We let $v(P, D, t)$, $P \in \{O, A, B, AB\} \cup \{\mathrm{NA}\}$ and $D \in \{O, A, B, AB\}$ as the set of nodes in $G(t)$ associated with a pair with patient's blood type P and donor's blood type D. Altruistic donors are modeled as pairs with an intended patient with blood type NA who can receive the kidney of any donor. We define directed cycle $C = (N_c, E_c)$ as a graph such that $N_c = \{v_1, \ldots, v_l\}$ and $E_c = \{(v_i, v_{i+1}) : \forall i \in 1, \ldots, l-1\} \cup \{(v_l, v_1)\}$. A cycle C belongs to a matching structure $P_1D_1, \ldots, P_lD_l$ if $v_i \in v(P_i, D_i)$ for all

ID	Description	ID	Description	ID	Description
1	AA-AA	2	BB-BB	3	ABAB-ABAB
4	OA-AO	5	BA-AB	6	BAB-ABB
7	BO-OB	8	BO-OA-AB	9	AAB-ABO-OA
10	ABA-AAB	11	ABO-OAB-ABB	12	BAB-ABB-BB
13	AAB-ABA-AA	14	ABO-OAB-ABO-OO-OAB	15	BO-OB-BB
16	OAB-ABA-AO	17	OO-OO	18	AO-OB-BA
19	ABO-OB-BB	20	OA-AA-ABO	21	BB-BO
22	ABO-OAB	23	AO-AO	24	BO-BO
25	ABA-ABA	26	ABB-ABB	27	BA-AO
28	OO-OA-AO-OB-BO	29	OO-OA-AO	30	OB-BO-OO
31	OA-AO-OO-OB-BO				

Table 3.6: Some matching structures used in our matching algorithms

$i = 1, \ldots, l$. Finally, for a matching structure S we define the compatibility graph $G(S, t)$ as the subgraph of $G(t)$ in which all cycles belongs to the matching structure $P_1D_1, \ldots, P_lD_l$. In Table 3.6 we specify matching structures that will be used to define some of our matching algorithms. The matching structures with ID 4, 7 can trivially be extended to have length 4. Similarly, matching structure with ID 17 can be extended to have length 3 or 5. In addition to the matching structures defined in Table 3.6, we define the matching structure with ID 32 as the set of matching structures in which all pairs' patients have AB blood type. Similarly, we define the matching structure with ID 33 as the set of matching structures in which all pairs' have patient blood type O or donor blood type O. Let $A(t)$ be the set of altruistic donors at time t.

Next we define some sets of matching structures $ID's$ that will be used to describe our algorithms:

$$
\begin{aligned}
S_{2,3} &= [3, 11, 6, 12, 15, 2, 9, 10, 13, 16, 14, 5, 7, 8, 1, 4, 17, 18, 20, 27, 23, 24, 25, 26, 21, 22, 32, 19, 29, 30], \\
S_{2,3,5} &= [3, 11, 6, 12, 15, 2, 9, 10, 13, 16, 14, 5, 7, 8, 1, 4, 17, 18, 20, 25, 26, 22, 19, 29, 30, 28, 31, 33].
\end{aligned}
$$

We proceed to describe the matching algorithms used in this work. For a given matching frequency δ and a maximum time horizon T, we define the i^{th} component of the time vector $\vec{t}$ as $(i-1)\,\delta$. For an arbitrary $t \in T$ and for a set of altruistic donors $A(t)$,

Matching algorithm 1

Step 1. Let $S_{\text{ID}} = S_{2,3}$.

Step 2. For $j = 1 : \text{length}(\text{S}_{\text{ID}})$, Find a circulation f^* of $G\left(S(j), t\right)$ that maximizes the number of edges that carries non-zero flow and such that the flow entering a node is at most one. For every edge $(v, u) \in E(t)$, if $f^*_{u,v}$ is non zero, update the status of the pairs associated with nodes v and u as matched and update $G(t)$. Otherwise, do nothing.

Step 3. If the set of altruistic donors is not empty, $A(t) \neq \emptyset$. For every altruistic donor $a \in A(t)$, find the longest shortest path from the node associated with the altruistic donor a to all nodes in $G(t)$ associated with patient-donor pairs that are reachable from the altruistic donor's node under consideration. Update $A = A \backslash \{a\}$, update the status of all nodes in the selected path as matched, and update $G(t)$. Otherwise, STOP.

Matching algorithm 2

Step 1. Randomly rearrange components in vector $S_{2,3}$. Let $S_{\text{ID}} = S_{2,3}$.

Step 2. For $j = 1 : \text{length}(\text{S}_{\text{ID}})$, Find a circulation f^* of $G\left(S(j), t\right)$ that maximizes the number of edges that carries non-zero flow and such that the flow entering a node is at most one. For every edge $(v, u) \in E(t)$, if $f^*_{u,v}$ is non zero, update the status of the pairs associated with nodes v and u as matched and update $G(t)$. Otherwise, do nothing.

Step 3. If the set of altruistic donors is not empty, $A(t) \neq \emptyset$. For every altruistic donor $a \in A(t)$, find the longest shortest path from the node associated with the altruistic donor a to all nodes in $G(t)$ associated with patient-donor pairs that are reachable from the altruistic donor's node under consideration. Update $A = A \backslash \{a\}$, update the status of all nodes in the selected path as matched, and update $G(t)$. Otherwise, STOP.

Matching algorithm 3

Step 1. Let $S_{\text{ID}} = S_{2,3,5}$.

Step 2. For $j = 1 : \text{length}(\text{S}_{\text{ID}})$, Find a circulation f^* of $G\left(S(j), t\right)$ that maximizes the number of edges that carries non-zero flow and such that the flow entering a node is at most one. For

every edge $(v,u) \in E(t)$, if $f^*_{u,v}$ is non zero, update the status of the pairs associated with nodes v and u as matched and update $G(t)$. Otherwise, do nothing.

Step 3. If the set of altruistic donors is not empty($A(t) \neq \emptyset$): For every altruistic donor $a \in A(t)$, find the longest shortest path from the node associated with the altruistic donor a to all nodes in $G(t)$ associated with patient-donor pairs that are reachable from the altruistic donor's node under consideration. Update $A = A \setminus \{a\}$, update the status of all nodes in the selected path as matched, and update $G(t)$. Otherwise, STOP.

Matching algorithm 4

Step 1. Randomly rearrange components in vector $S_{2,3,5}$. Let $S_{\mathrm{ID}} = S_{2,3}$.

Step 2. For $j = 1 : \mathrm{length}(\mathrm{S_{ID}})$, Find a circulation f^* of $G\left(S(j),t\right)$ that maximizes the number of edges that carries non-zero flow and such that the flow entering a node is at most one. For every edge $(v,u) \in E(t)$, if $f^*_{u,v}$ is non zero, update the status of the pairs associated with nodes v and u as matched and update $G(t)$. Otherwise, do nothing.

Step 3. If the set of altruistic donors is not empty($A(t) \neq \emptyset$): For every altruistic donor $a \in A(t)$, find the longest shortest path from the node associated with the altruistic donor a to all nodes in $G(t)$ associated with patient-donor pairs that are reachable from the altruistic donor's node under consideration. Update $A = A \setminus \{a\}$, update the status of all nodes in the selected path as matched, and update $G(t)$. Otherwise, STOP.

Unconstrained

Step 1. Find a circulation f^* of $G\left(t\right)$ that maximizes the number of edges that carries non-zero flow. For every edge $(v,u) \in E(t)$, if $f^*_{u,v}$ is non zero, update the status of the pairs associated with nodes v and u as matched and update $G(t)$. Otherwise, do nothing.

3.3 The Experiments

In this work we study the scenario in which there is a national unified exchange program. Specifically, all pairs that are willing to participate in an exchange are registered in a centralize clearinghouse that run the matchings over time under different matching mechanisms. We consider a

(a) Pair's patient blood type

Patient's BT	Probability
O	49.07%
A	32.44%
B	14.66%
AB	3.83%

(b) Pair's donor blood type

Donor's BT	Probability
O	65.52%
A	26.26%
B	7.47%
AB	0.75%

Table 3.7: Distribution of blood types of patients and donors in the stream of arrivals

variety of matching mechanisms parametrized by: (i) presence of altruistic donors; (ii) the participation of compatible patient-donor pairs; (iii) allowed matching structures (cycles or chains); (iv) length of the allowed structures; and (v) the frequency with which matchings are run. In particular, the analysis of these features establishes an overarching link between the dynamics of the kidney exchange market and parameters under the control of policy makers and healthcare administrators. We assess the mechanisms by their long-run efficiency and waiting times on the 48 ABO-PRA pairs types considered in the model. Each scenario is run by matching pairs every 5, 10, 15 or 30 days, over a time horizon of 10 years. To generate the pairs, we use the same cPRA distribution, patient's and donor's blood type distribution as well as departure rate across all the scenarios, these are shown in Tables 3.4, 3.7 and 3.3.

In the Table 3.9 we present the conditional distribution of the donors blood type given the patient's blood type and its cPRA and in Table 3.10 we show the donor's blood type distribution when compatible pairs do not join the exchange pool.

3.4 Our findings

In this section we present our results in the context of three central questions: What is the probability of receiving an offer? How frequently should we run the matching mechanism? And who benefits when chains and longer cycles are allowed? We also discuss the sensitization level of the exchange pool over time. For each scenario in Table 3.8 we run 10 realizations. For each realization we run the matching algorithm every 5, 10, 15 and 30 days during 9 years and study the behavior

Scenario	**Matching algorithm**	**Description**
1	1	No altruistic donors, compatible pairs join the pool.
2	2	No altruistic donors, compatible pairs join the pool.
3	unconstrained	No altruistic donors, compatible pairs join the pool.
1a	1	Altruistic donors, compatible pairs join the pool.
2a	2	Altruistic donors, compatible pairs join the pool.
4	3	No altruistic donors, compatible pairs don't join the pool.
5	4	No altruistic donors, compatible pairs don't join the pool.
6	unconstrained	No altruistic donors, compatible pairs don't join the pool.
4a	3	Altruistic donors, compatible pairs don't join the pool.
5a	4	Altruistic donors, compatible pairs don't join the pool.

Table 3.8: Description scenarios recreated in our experiments

(a) Pair's with patient's cPRA value 0.075

D / P	O	A	B	AB
O	0.1247	0.3254	0.1513	0.6552
A	0.6667	0.1304	0.8084	0.2626
B	0.1896	0.4946	0.0172	0.0747
AB	0.019	0.0496	0.0231	0.0075

(b) Pair's with patient's cPRA value 0.625

D / P	O	A	B	AB
O	0.542	0.624	0.564	0.6552
A	0.348	0.2502	0.362	0.2626
B	0.1	0.1144	0.064	0.0747
AB	0.01	0.0114	0.01	0.0075

(c) Pair's with patient's cPRA value 0.95

D / P	O	A	B	AB
O	0.6435	0.652	0.646	0.6552
A	0.2715	0.262	0.272	0.2626
B	0.0.077	0.078	0.074	0.0747
AB	0.008	0.008	0.008	0.0075

Table 3.9: Conditional distribution of donor's blood type in the stream of arrivals given the patient's blood type and its cPRA .

Donor's BT	Probability
O	0.397
A	0.403
B	0.18
AB	0.02

Table 3.10: Donor's blood type distribution when compatible pairs don't join the pool.

overtime of 1) the number of pairs waiting over time in the pool 2) the number of pairs that receive an offer 3) the number of pairs that depart after waiting one year or more and 4) the waiting time distribution of pairs that receive an offer. The standard deviations of the provided measures are not reported in this document but can be found in the Excel technical report associated with this work. To measure the performance of policies regarding departures, we don't penalize a policy for pairs that depart without an offer within one year of their arrival. Fielded exchange programs have average waiting times of at most 18 months depending on the different ABO-PRA pair types, and thus we assume that a policy failed a patient only when the patient departs without an offer after having waited more than one year.

3.4.1 How frequently should we run the matching algorithms?

We measure the effect of the frequency at which the matchings are run on the long-term efficiency across the different ABO-PRA pairs' types in our model, as well as on their waiting times.

Compatible pairs join the pool In *Scenario1*, *Scenario1a* and *Scenario3* the differences in the performances for low and medium sensitized patients for the different matching frequencies are negligible. Moreover, more than 95% of low and medium sensitized patients receive an offer within one year of their arrival independent of type of their donor; this result is not surprising. As expressed by the attendees of the symposium "Future Directions in Paired Exchange"[4] in 2013, "all KPD[5] programs struggle to transplant highly sensitized recipient candidates (cPRA>98%)"[6]. Thus, we will discuss our findings for the highly sensitized patients. The results for the low and medium sensitized patients can be found in the Excel technical report.

Our findings for *Scenario1* when running matchings every 5 days, 10 days 15 days or 30 days suggest that in the long-run, the frequency at which matchings are run has only a slight impact on the efficiency across the different ABO-PRA pair types. The waiting times are in average shorter

[4]Continuing Medical Education event organized by the Weill Cornell Medical College, April 19th 2013.

[5]Kidney Paired Donation

[6]Adam Bingaman, Director of the Live Donor and Incompatible Kidney Transplant Program at the Methodist Specialty and Transplant Hospital in San Antonio ,TX

when matching every 5 days than when matching every 30 days — recall that we are discussing highly sensitized patients. The patient's waiting time cumulative density function obtained when matching every 5 days is lower bounded by the one obtained matching every 30 days with few exceptions, as pairs with a B patient and an A or B donor. Specifically, our findings suggest that matching every month instead of every 5 days does not increase the number of pairs that receive a matching offer neither reduce the time that pairs attend in the exchange program before receiving an offer in *Scenario1*. To illustrate this observation, in Table 3.11 we show the cumulative density function of pairs with an O highly sensitized patient and a B donor in *Scenario1*.

	5 days	**10 days**	**15 days**	**30 days**
6m	73.59%	70.07%	70.27%	70.00%
12m	94.38%	93.19%	91.65%	91.22%
18m	99.51%	98.30%	98.28%	98.78%
2y	100.00%	99.27%	99.51%	99.76%
2.5y	100.00%	100.00%	100.00%	100.00%
3y	100.00%	100.00%	100.00%	100.00%

Table 3.11: Cumulative density function of the waiting time of pairs with an O highly sensitized patient and an B donor that receive an offer in *Scenario1*

About efficiency performance across the different type of pairs with a highly sensitized patient, in *Scenario1* at least 95% of O and A patients receive an offer, and at least 85% of all B and AB patients do. Given that pairs with an AB patient are blood compatible with any donor and given their low rate of arrival, in algoritm 1 there are a limited set of structures through which pairs with an AB patient are matched. This latter is hurting the number of pairs with an AB patient that receive an offer.

For *Scenario3* we found that the frequency at which matchings are run has negligible impact in the number of pairs that receive an offer but for pairs with an O patient and an AB donor; for which matching less frequently significantly improve their efficiency and waiting times. For pairs with a non AB donor *Scenario3* offers a transplant to at at least 95% of the pairs that arrive to the program while only 59% of the pairs with an O patient and an AB donor receive an offer. Even

though in overall in *Scenario3* more offers are made, the disparity in the performance due to the donor's blood type is larger than in *Scenario1* and *Scenario1a*.

About waiting times in *Scenario1*, *Scenario1a* and *Scenario3*, for all pair types but pairs with an AB donor 90% of the pairs that receive an offer do it within 12 months of arrival and at least 95% does it within 18 months of arrival. However in *Scenario1* at least 90% of the pairs with an AB donor receive an offer within 18 months of arrival while in *Scenario1a* this percentage drops to 82.5% when matchings are run every 5 days and is at least 90% when matchings are run every 10, 15 and 30 days and in *Scenario3* it drops to 56% when running every 5 days and is at least 75% when running every 10, 15 and 30 days.

In conclusion, contrary to the static model, our findings suggest that running matching less frequently doesn't provide a significant improvement in efficiency. While in *Scenario1* and *Scenario1a* running more frequently offers smaller waiting times for those pairs that receive an offer, we observe the opposite behavior in *Scenario3*, in which matching less frequently result in smaller waiting times for highly sensitized patients.

Due to the disparity in the quantities of the different ABO-PRA types in consideration that arrives to the pool, measures across all patients with the same blood type and cPRA level hide the differences in performance due to their particular donor blood type. In Tables 3.12 we present the percentage of pairs with O or A highly sensitized patient by donor's blood type that receive an offer. The results for pairs with patients with blood type B or AB as well as the number of offers and other details can be found in the technical report in Excel.

Compatible pairs don't join the pool From the donors blood type distribution in Table 3.10 and the patient's blood type distribution in Table 3.7 it follows that in the stream of arrivals there is a large disparity between the supply and the demand of blood type O kidneys. Meanwhile 49.07% of the patients that arrive to the pool have blood type O, only 39.7% of the donors do. As consequence is expected that a considerable percentage of blood type O patients depart from the exchange program without an offer.

In contrast to the case when compatible donors join the pool, in *Scenario4*, *Scenario4a* and *Scenario6* the performance across patients with low and medium cPRA varies depending on their

		Scenario3				*Scenario1*			
Donor BT	**Patient BT**	**5 days**	**10 days**	**15 days**	**30 days**	**5 days**	**10 days**	**15 days**	**30 days**
O	O	99.11%	99.14%	99.24%	99.32%	98.81%	98.91%	98.91%	98.99%
O	A	99.55%	99.55%	99.57%	99.64%	99.11%	98.58%	98.74%	98.53%
A	O	98.25%	98.53%	98.53%	98.86%	97.90%	97.89%	97.96%	98.10%
A	A	99.22%	99.40%	99.46%	99.46%	90.75%	91.20%	90.98%	90.54%
B	O	98.50%	99.31%	98.77%	99.32%	97.85%	97.16%	96.90%	96.47%
B	A	99.18%	99.79%	99.38%	99.59%	95.91%	97.41%	95.24%	97.79%
AB	O	**59.26%**	**83.87%**	**77.05%**	**92.31%**	**97.83%**	**96.08%**	**95.92%**	**95.92%**
AB	A	88.68%	92.73%	92.45%	96.15%	85.19%	89.29%	89.66%	89.66%

Table 3.12: Percentage of highly sensitized O and A patients by donor that receive an offer in *Scenario1* and *Scenario3*

donor's blood type . The difference in the number of pairs that receive an offer as a function of the frequency at which matchings are run is meaningful mainly for non O blood type patients with a non O donors. We will focus our discussion on pairs with a non blood type O donor. In Table 3.13, 3.14 and 3.15 we show the percentage of O patients with a non O donor that receive an offer in *Scenario4*, *Scenario4a* and *Scenario6* for low, medium and highly sensitized patients respectively when matching every 5 days and every 30 days.

	Scenario6			*Scenario4a*			*Scenario4*		
Frequency	**A**	**B**	**AB**	**A**	**B**	**AB**	**A**	**B**	**AB**
5 days	87.28%	80.37%	**10.36%**	79.37%	81.25%	89.81%	77.45%	81.02%	93.65%
30 days	86.13%	78.63%	**5.18%**	74.65%	80.10%	93.71%	71.86%	80.53%	96.03%

Table 3.13: Percentage of patients with an O low sensitized patient that receive an offer classified by their's donor's blood type

While slightly more O low sensitized patients with a non O donor receive an offer when matching more frequently (every 5 days), more O medium and highly sensitized patients with a non O donor receive an offer when matchings are run every 30 days in *Scenario4*, *Scenario4a* and *Scenario6*. The percentage differences are larger as the cPRA level increases. For example when matching every 5

	Scenario6			*Scenario4a*			*Scenario4*		
Frequency	**A**	**B**	**AB**	**A**	**B**	**AB**	**A**	**B**	**AB**
5 days	74.75%	62.50%	13.89%	81.59%	70.37%	80.95%	83.42%	72.00%	79.87%
30 days	77.78%	73.91%	11.11%	90.85%	67.74%	85.71%	90.83%	73.01%	86.90%

Table 3.14: Percentage of patients with an O medium sensitized patient that receive an offer classified by their's donor's blood type

	Scenario6			*Scenario4a*			*Scenario4*		
Frequency	**A**	**B**	**AB**	**A**	**B**	**AB**	**A**	**B**	**AB**
5 days	25.58%	28.69%	0.00%	62.13%	36.96%	31.82%	56.78%	32.18%	62.44%
30 days	32.12%	35.05%	5.36%	73.37%	37.21%	60.87%	71.85%	32.10%	67.96%

Table 3.15: Percentage of patients with an O medium sensitized patient that receive an offer classified by their's donor's blood type

days in *Scenario4* 56.78% of the highly sensitized patients that have arrived by the year 9 with an O patient and A donor receive an offer, 71.85% does when matching every 30 days. For pairs with medium sensitization level the relation is 83.42% to 90.83% and for pairs with low sensitization level it goes from 77.45% to 71.86%. This suggest that the effect of the frequency at which matchings are run is more significant for highly sensitized pairs and that running less frequently does benefit these pairs on detrimental of pairs with a low sensitized patient. Notice also that highly sensitized O patients with an AB donor receive significantly more offers in *Scenario4* than in *Scenario4a* and *Scenario6* when chains and unconstrained length cycles are allowed. Also, a meaningful additional percentage of medium sensitized O patients with an AB donor receive an offer in *Scenario4* and *Scenario4a* than in *Scenario6*.

The frequency at which matchings are run also impact the distribution of the waiting times for pairs with an O patient in all the scenarios in consideration, and do not have significant impact for pairs with non O patient which are low and medium sensitized, because these latter receive an offer in average within 12 months of their arrival for all the frequencies and matching algorithms in consideration with few exceptions as pairs with an AB donor and an AB patient which are scarce.

We found that the percentage of pairs with an O patient that receive an offer and do it within 12 months of their arrival is larger when matching every 30 days than when matching every 5 days. In Table 3.16 we present the waiting time cumulative density function of pairs with an O highly sensitized patient and an A donor.

	Scenario6				*Scenario4a*				*Scenario4*			
	5 days	10 days	15 days	30 days	5 days	10 days	15 days	30 days	5 days	10 days	15 days	30 days
6m	29.53%	30.49%	33.04%	27.24%	20.03%	26.11%	30.53%	40.58%	14.63%	21.75%	25.40%	37.2%
12m	47.67%	57.85%	56.52%	63.07%	33.79%	43.52%	49.50%	64.68%	27.95%	38.45%	44.90%	58.88%
18m	64.19%	71.75%	75.00%	80.80%	49.24%	58.53%	65.18%	81.02%	43.90%	54.17%	62.08%	74.97%
2y	77.21%	84.08%	84.35%	89.21%	65.37%	75.09%	77.72%	89.89%	60.98%	71.26%	76.39%	86.95%
2.5y	86.28%	91.26%	93.48%	94.88%	80.65%	85.84%	88.94%	95.43%	78.05%	83.88%	87.48%	94.41%
3y	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%

Table 3.16: Waiting time cumulative density function for pairs with an O highly sensitized patient and an A donor

3.4.2 What are the odds to receive an offer?

Besides the natural motivation that arises from the concern of many practitioners and exchange programs of the inherent difficulty of some pairs to participate in an exchange due to a highly sensitized patient or a donor with a under demanded blood type, gaining insights on how policies impact the odds of a pairs to receive an offer would help us to understand the success of small single center programs whose decisions of offering or not a transplant to a patient heavily relies in the use of practitioners knowledge rather than on the maximization of overall efficiency; small exchange programs generally match pairs through short cycles and rarely through chains. We model the practitioner's knowledge in the decision process through the matching structures define in Table 3.6.

Let X_{d} be a set of donor's blood type, X_{p} a set of patient's blood types and S a set of patient's cPRA levels: high, medium or low. Let $P(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t)$, be the number of pending pairs at time t with donor blood type in set X_{d}, patient's blood time and cPRA in sets X_{p} and S respectively. Similarly, $M(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t)$ and $P(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t)$ denote the number of matched and departed pairs

that attended at least one year by time t with donor blood type in set X_{d}, patient's blood time and cPRA in sets X_{p} and S respectively. We define the percentage of matched pairs for a set $X_{\mathrm{d}}, X_{\mathrm{p}}, S$ as

$$p_{\text{eff}}\left(X_{\mathrm{d}}, X_{\mathrm{p}}, S\right) \quad = \quad \left.\frac{M(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t)}{M(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t) + D(X_{\mathrm{d}}, X_{\mathrm{p}}, S, t) + D(X_{\mathrm{d}}, X_{\mathrm{d}}, S, t)}\right|_{t=9 \text{ years}}.$$

We define $p_{\text{eff}}\left(X_{\mathrm{d}}, X_{\mathrm{p}}, S\right)$ as the *odds* that pairs with patients blood type in X_{p}, cPRA classification in S and donor's blood type in X_{d} receive an offer when t goes large.

Compatible pairs join the pool As discussed in Section 3.4.1, the odds to receive an offer for low and medium sensitized patients in *Scenario1*, *Scenario1a* and *Scenario3* are at least 95% independently of their willing donor's blood type, thus we focus our discussion on pairs with a highly sensitize patient. The odds to receive an offer for a patient randomly selected are at least 85% in *Scenario1* and at least 90% in *Scenario1a* and *Scenario3*. However the large disparity in the number of pairs with the same blood type but different donor's blood type hides large differences in the odds for highly sensitize patients with the same blood type to receive an offer depending on their willing donor's blood type.

Meanwhile allowing pairs to be matched through cycles of arbitrary length it roughly increases by 2% the odds to receive an offer for O and A highly sensitize patients but the ones with an AB willing donor, it significantly reduces the odds to receive an offer for pairs with an AB donors and patient with blood type O or A, which drops from 97% in *Scenario1* to 59% in *Scenario3* when running matchings every 5 days respectively, see Table 3.12. In other words, matching through small cycles we reduce the disparity in the odds to be matched across the different ABO pair types in consideration for highly sensitized patients at expenses of a 2% drop in the odds to receive an offer for pairs with a non AB donor. Notice that the odds obtained for AB patients in *Scenario1* are significantly smaller than in *Scenario1a* and *Scenario3*. The reason behind this result is the limited set of structures used in matching algorithms 1 in which pairs with an AB patient are included, we do so due to their inherent ease to receive an offer.

Compatible pairs don't join the pool Our results suggest that the odds to receive an offer for pairs with an non O patient are at least 80% in *Scenario4* as well as in *Scenario6*. Unlike the results obtained in *Scenario6* in which the odds to receive an offer for pairs with a non O patient

are at least 95% for all cPRA levels, in *Scenario4* and *Scenario4a* the cPRA level of the pair has a larger impact on the odds for pairs with a non O patient to receive an offer and it drops from 89% to 83% for pairs with an A low and medium sensitized patient respectively.

The increase in the odds to receive an offer obtained in *Scenario6* for non O patients contrasts with the odds to receive an offer obtained for pairs with an O patient. Meanwhile in *Scenario6* at least 80% of the O low sensitized patients with an O, A or B donor receive an offer, at most 12% of those with an AB donor receive one. This dissacords with the results in *Scenario4*, in which at least 75% of the O low sensitized patients receive an offer for each donor's blood type, and at least 80% of the low sensitized patients receive an offer in overall. This disparity in the likelihood of the O low sensitized patients to receive an offer extends to medium and low sensitized O patients as well as to all pairs with non O donor. In summary our findings suggest that when compatible pair's don't participate in the exchanges matching through small cycles we can potentially reduce the disparity in performance across different pairs in the model, while still offering acceptable levels of odds to receive an offer. Not surprising the results in efficiency reflect on the waiting time distributions. Meanwhile the majority of non O patients that receive an offer do it within 12 months of their arrival, this proportion drops to 50% or less for O patients with a non O donor in both *Scenario4* and *Scenario6* across for highly sensitized patients. Significant drops in the proportions of pairs of pairs that receive an offer within 12 months of their arrival also occur for medium and low sensitized patients.

3.4.3 How is the sensitization level of the pool?

One of the main arguments that supports the call for long chains is that the exchange pools that we are observing in practice are mainly composed by pairs with highly sensitized patients and given that chains would lift the tight coincidence requirements imposed by small cycles, more pairs could benefit through the allowance of chains. In this Section we discuss about the sensitization level of the pairs in the pool that results from our model and how it differs from the sensitization level of the stream of arrival in Table 3.4.

Compatible pairs join the pool For all the ABO-PRA pair types in the model, the composition of the pool is stable after 9 years of been running in *Scenario1*, *Scenario1a* as well as in *Scenario3*.

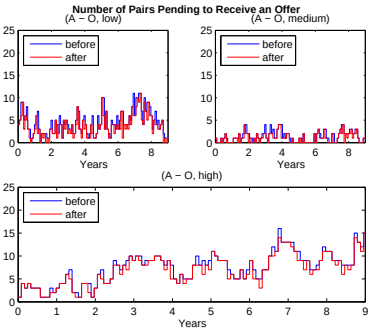
However the proportion of low sensitized and high sensitized patients behave differently in these two scenarios. In *Scenario3* we encounter a pool for which for all ABO pair types but pairs with an AB donor the proportion of highly sensitized patients is at most 50%. However in *Scenario1* we encounter a pool for which at most 50% of the patients are low sensitized. Thus, we observe a more sensitize pool when matching through small cycles than when allowing arbitrary length cycles. In Table 3.17 we present the percentage of each ABO pair type with donor's blood type O, A and B that have a highly sensitized patient by the end of the 9th year the exchange program is running. In this table we can observe that for the different ABO pairs type for which we display information, the percentage of highly sensitized patients in *Scenario1* and *Scenario1a* is higher than in *Scenario3*, for instance, meanwhile 25% of the pairs with a B donor and a B patient in *Scenario3* pending by the end of the 9th year, this percentage rise to 85.71% and 70.59% in *Scenario1a* and *Scenario1* respectively when matchings are run every 30 days.

	Scenario3			*Scenario1a*			*Scenario1*		
Patient BT	**O donor**	**A donor**	**B donor**	**O donor**	**A donor**	**B donor**	**O donor**	**A donor**	**B donor**
O	24.71%	35.37%	23.81%	33.57%	37.10%	41.67%	28.57%	41.43%	52.00%
A	17.05%	16.67%	22.22%	37.50%	**63.16%**	**70.00%**	39.13%	**70.65%**	46.15%
B	25.49%	20.00%	25.00%	28.89%	39.77%	85.71%	30.43%	38.24%	**70.59%**
AB	12.50%	14.29%	0.00%	**53.85%**	**52.94%**	**50.00%**	**50.00%**	**52.17%**	**87.50%**

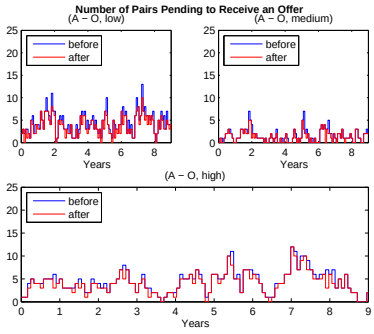
Table 3.17: Percentage of the different ABO pair types with a highly sensitized patient pending to receive an offer by the end of the 9th year when compatible pairs join the pool and matchings are run every 30 days

In both *Scenario1* and *Scenario3* the cPRA distribution found in the pool varies significantly from the cPRA distribution in the stream of arrivals. Even though the pool observed in *Scenario1* is more sensitized, it results in a more equilibrate solution in terms of odds to receive an offer for all ABO-PRA pairs in consideration while providing acceptable waiting times distribution. This observation could be explained by the fact that the size of the pool at an arbitrary point on time is larger in *Scenario1* than in *Scenario3* however the extra burden introduce by higher sensitization level seems to be compensated by the larger size of the pool. In figure 3.4.1 and figure 3.4.2 we show the number of pairs with O patient and A donor over 9 years and the number of pairs with

A donor and B patient over 14 years respectively for the three sensitization levels in the model in *Scenario1* and *Scenario3* respectively just after and before running a matching when matching every 30 day.

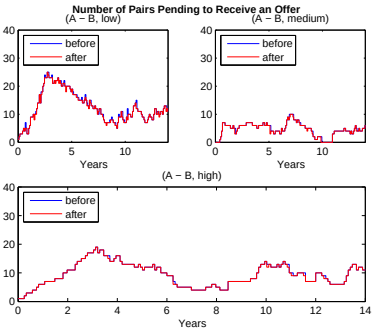


(a) *Scenario1*

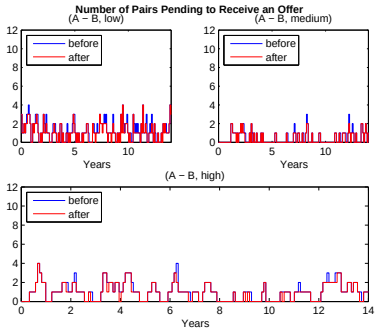


(b) *Scenario3*

Figure 3.4.1: Number of pairs with an O patient and an A donor pending to receive an offer over time in *Scenario1* and in *Scenario3* when matching every 30 days



(a) *Scenario1*



(b) *Scenario3*

Figure 3.4.2: Number of pairs with an B patient and an A donor pending to receive an offer over time in *Scenario1* and in *Scenario3* when matching every 30 days

In conclusion, when matching through unconstrained length cycles or cycles bounded by length 3 we observe a pool with a sensitization level higher than the one of the stream of arrivals. This difference is larger when matching through small cycles. However this latter results in a solution with smaller disparity in the likelihood of receiving an offer across the different ABO-PRA pair types. Our results suggest that rate of arrivals, disparity among the supply and demand of kidneys

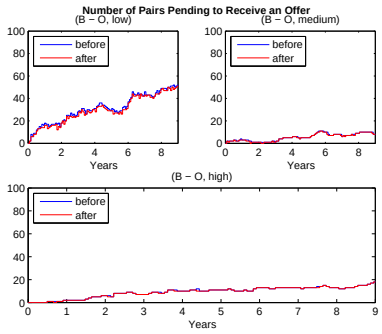
of the different blood types as well as the sensitization level play an important role in the success of a policy, and not only the level of sensitization of the pairs pending to be matched over time as it is more commonly modeled.

Compatible pairs don't join the pool Unlike the case when compatible pairs join the pool, the composition of the pool does not reach a steady state by the 9th years that is running; there are build ups of O low sensitized patients with non O donor as shown in Figure 3.4.3 for pairs with an B donor and an O patient. Contrary to what is expected from a pool that is difficult to clear even when the constraint on the length of the cycles is lifted, we found that for the different ABO pair types in consideration but the ones with an O donor, the proportion of low sensitized patients is at least 50%, thus we are observing a pool with lots of low sensitize patients. In Table 3.18 we present the proportion of each ABO pair type with donor's blood type O, A and B and highly sensitized patient by the end of the 9th year. This observation seems to be consequence of the rate of arrivals of pairs with low cPRA, 55%, together with the disparty of the demand and suply of blood type O kidneys which cause large build ups of pairs with O patient and non O donor that weaken the advantage of pairs with low cPRA due to a larger size pool. This is observed in *Scenario4*, *Scenario4a* as well as in *Scenario6*.

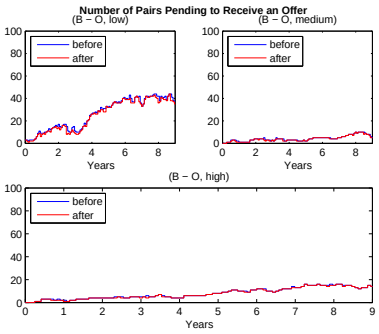
	Scenario6			*Scenario4a*			*Scenario4*		
Patient BT	**O donor**	**A donor**	**B donor**	**O donor**	**A donor**	**B donor**	**O donor**	**A donor**	**B donor**
O	**71.43%**	27.96%	25.41%	**82.55%**	13.03%	21.66%	**82.70%**	15.00%	20.93%
A	38.81%	22.73%	4.60%	51.72%	68.85%	9.16%	34.71%	78.15%	6.10%
B	61.11%	27.03%	66.67%	37.50%	23.08%	66.67%	44.59%	18.92%	100.00%
AB	33.33%	40.00%	50.00%	33.33%	50.00%	100.00%	14.89%	64.62%	79.38%

Table 3.18: Percentage of the different ABO pair types with a highly sensitized patient pending to receive an offer by the end of the 9th year when compatibles don't join the pool.

In conclusion we encounter a pending pool by the end of the 9th year with more than 50% of low sensitized patients for the different ABO pair types with an O patient however difficult to clear. This result extends to the pool in overall. The disparity in the supply and demand of kidneys of a particular blood type, in this case blood type O, seems to play a significant role in the difficult to clear the pool and affects the relevancy of the sensitization level of pairs in order to receive an



(a) *Scenario4*



(b) *Scenario6*

Figure 3.4.3: Number of pairs with an O patient and an B donor pending to receive an offer over time in *Scenario4* and in *Scenario6* when matching every 30 days for a realization

offer.

3.4.4 Who benefits from allowing chains and longer cycles?

Compatible pairs join the pool The overall increase in the numbers of patients that receive an offer for all ABO-PRA pairs when 1% of the arrivals are altruistic donors is 4%; however among highly sensitized patients this improvement is of at most 2% in average for all the matching frequencies.

The distribution of the bridge donors that terminate a chain have a pivotal importance in the ongoing debate about the pros and cons of deviating altruistic donors from the waiting list to the exchange programs and offer the last bridge donor in a chain to the waiting list. We found that for all the matching frequencies in consideration the blood type distribution of the bridge donors is significantly different from the one of the altruistic donor's. 67.7% of the altruistic donors have blood type O, only 12.08% of the bridge donors that terminate a chain have blood type O; also while 26% of the altruistic donors have blood type A 66% of the bridge donors have blood type A. In Table 3.19 we show the blood type distribution of the last bridge donor of the chains that were created in *Scenario3*. Recall that the expected altruistic donor's blood type distribution is equal to the willing donor's blood type distribution. Meanwhile 65% of the altruistic donors are expected to have blood type O, only 13% of the chains terminated with an O donor when bridge donors do not wait for the next run of the matching and the matchings are run every 5 days, this

percentage raise to 16% when matchings are run every 30 days. We are not aware of any work in the literature that measure the impact on patients in the waiting list of deviating altruistic donors from the waiting list to initiate chains in the exchange pool and this latter is an important element to understand what are the pros and cons of allowing chains initiated by altruistic donors.

Bridge Donor BT	**5 days**	**10 days**	**15 days**	**30 days**
O	12.08%	12.92%	15.77%	16.06%
A	66.29%	66.85%	63.10%	71.55%
B	15.17%	15.17%	14.08%	9.58%
AB	6.46%	5.06%	7.04%	2.82%

Table 3.19: Chain's last bridge donor blood type distribution in *Scenario1a*

Compatible pairs don't join the pool Unlike the case in which compatible pairs join the pool, our results suggest that when 1% of the arrivals are altruistic donors the overall improvemet in the number of offers is negligible, in *Scenario4a* we observe an improvement in the number of offers respect to *Scenario4* of at most 1% for the different matching frequencies in consideration. What result surprising is that among highly sensized patients chains did not increase the total number of offers, and in fact there is a decrease of 1.6% when matching are run every 5 days and of 1.14% when matchings are run every 30 days. However it is important to recall that in *Scenario4* the pair's rate of arrival is 1% less than its rate of arrivals in *Scenario4* because 1% of the arrivals are altruistic donors. The donor's rate of arrivals is the same in *Scenario4* and in *Scenario4a*. These latter facts together with the fact that the impact of the rate of arrivals on the performance of the exchange program is not linear does not allows to conclude that chains actually reduce the number of offers to highly sensitized patients while increase by a small margin the overall number of offers. However these findings suggest that the improvement in the number of offers seems to be marginal whe compatible pairs don't enter the pool and thus reinforce the call to measure the actual impact of desviating altruistic donors from the waiting list to initiate a chain in the exchange program.

Among pairs that receive an offer, pairs with a low sensitized patient increase their expected waiting times when chains are allowed while pairs with a highly sensitized patient reduce their

waiting times. In Tables 3.20 and 3.21 we show the waiting time's cumulative density function of pairs with an A donor and an O highly sensitized patient and low sensitized patient respectively when matchings are run every 5 days and every 20 days.

	Scenario4a		*Scenario4*	
	5 days	**30 days**	**5 days**	**30 days**
6m	20.03%	40.58%	14.63%	37.20%
12m	33.79%	64.68%	27.95%	58.88%
18m	49.24%	81.02%	43.90%	74.97%
2y	65.37%	89.89%	60.98%	86.85%
2.5y	80.65%	95.43%	78.05%	94.41%
3y	100.00%	100.00%	100.00%	100.00%

Table 3.20: Pairs with an O highly sensitized patient and an A donor Waiting time's cumulative density function

	Scenario4a		*Scenario4*	
	5 days	**30 days**	**5 days**	**30 days**
6m	29.31%	36.15%	28.67%	33.92%
12m	49.73%	54.90%	49.65%	53.91%
18m	70.66%	69.89%	65.48%	68.36%
2y	84.85%	81.01%	78.84%	78.97%
2.5y	95.30%	89.55%	91.11%	87.78%
3y	100.00%	100.00%	100.00%	100.00%

Table 3.21: Pairs with an O low sensitized patient and an A donor Waiting time's cumulative density function

In Conclusion, for pairs with an O patient, more low sensitized patients receive an offer when chains are allowed however among those that receive an offer the expected time to receive an offer increases, for pairs with a low sensitized patient, the increase in the number of offers is minimal

however pairs that receive an offer does faster than when chains are not allowed.

3.5 Discussion

We have examined the impact of the frequency at which matchings are run, the length and type of the structures through which matching can be performed -cycles or chains and the will of compatible pairs to participate in an exchange with another pair on the number of first time offers that are done, the waiting time to receive an offer as well as the level of sensitization of the observed pool across 48 ABO-PRA pairs types considered in our model. These features of the kidney exchange program are well understood in the static set up and some theoretical models as well as empirical work have been done in the dynamic set up, mainly on the structure that are used and overall performance measures. In this work we identified key features in the design of policies to run kidney exchange programs and analyze their long term impact on the 1) number of pairs that receive an offer 2) expected waiting time to receive an offer 3) sensitization level of the observed pool and evaluated performances across the different ABO-PRA pair types instead of overall measures 4) bridge donors that terminate chains blood type distribution.

The main contribution of this work is that it provides insights on several central questions in this subject. Our findings suggest that the on going debate on the unified call for the use of long chains and their benefit on the different type of pairs in the exchange pool classified by their donor's blood type, patient's blood type and sensitization level should be more carefully consider, while in overall it brings improvement in efficiency it does so increasing the number of offers to pairs with low sensitization level instead of pairs with high sensitization level when compatible pairs do not join the exchange program. When compatible pairs join the pool, the overall improvement brought by chains is larger than in the case in which compatible pairs don't participate in exchanges with other pairs and there is is not clear evidence to support the idea that pairs with highly sensitized patients benefit meaningfully. The frequency at which matchings should be run does not seems to be a key differentiator element when compatible pairs join the exchange pool however it has a substantial effect when only incompatible pairs join the pool. In this latter case the effect of the frequency at which pairs are matched varies across the different type of pairs, in other words, there is a trade off across the different type of pairs when matching more or less frequently. The difficulty to clear a

pool is usually associated and modeled with the proportion of pairs with a highly sensitize patient. Our findings suggest that even though this latter is of central importance, the large disparity in the supply and demand of kidneys of the different blood types is a prime factor to determine the success of a particular policy and the needs of different type of structures - longer cycles or chains to match the pairs. In particular, we found a more sensitized pool prior to a matching when compatible pairs join the pool, however the performance across the different pair types -efficiency and waiting times obtained for the different policies in these scenarios were better than when compatible pairs did not join the pool, and the policies in consideration were having difficulty even though the pool seen by a central planer prior to a matching was less sensitized. Finally our findings also indicate that when allowing larger cycles or chains larger disparity in number of offer and waiting times arise across the different type of pairs however in overall there is improvement in the number of pairs and expected waiting times. Particularly, when compatible pairs join the pool, pairs with an AB donor are significantly hurt when cycles length are unconstrained. In the case that compatible pairs do not join the pool, pairs with an O patient are significantly less likely to receive an offer and among those that receive an offer, waiting times are ostensible longer than for the pairs with non O patients. These differences are accentuated as the level of cPRA increases.

Another contribution of this work is the evaluation of policies through worst performance measures across the different type of pairs in the model. Specifically, we use the notion of minimizing of the disparity in performance across the different type of pairs as a response to a concern raised by some practitioners and exchange programs staffs that consider that the actual challenge of the programs should not focus in the ability to match those pairs that any policy or program could match but identifying and help to match those pairs that have an inherent disadvantage to receive an offer under the majority of policies currently running fielded exchanges programs. Due to the disparity in the numbers of the different pairs blood types that arrive to an exchange program overall measures hide serious under performance for some pair types.

We acknowledge that our study have several limitations, which in consequently suggest multiple future research directions. First we did not have access to detailed empirical data about patients that joined exchange programs from fielded exchange programs, in particular it would be interesting to have data from different sizes exchange programs which are actually running under different policies to validate our findings. Second a sensitivity analysis on the rate at which pairs join the

pool would help to answer questions as: is there a threshold rate of arrivals after which relative improvements are minimal? could we achieve similar level of performances with multiple regional exchange programs and avoid the burden of inter institutional nationwide logistics instead of an unified national one?, which will provide further insights in the success of small programs, as the one directed by Adam Bingaman at the Methodist Specialty Hospital in San Antonio,Texas. At last, in this work we focus on the odds of a pair to receive a first offer after arrival and how it is affected by the different features involve in the design of a policy. We understand that a flaw of this model is the assumption that pairs departed after the first offer after their arrival to the exchange program, however it provides insights in the inherent difficulty that is measured by cPRA level as well as blood type, which is the preliminary information used to consider the possibility of conducting further tests that may result in an actual transplant offer. In practice, making an offer is by far no guarantee of an actual execution of a transplant an actually the reported ratios of offers and actual transplants reported by field exchange programs are somehow worrisome. It is central to understand the impact on the performance of a nonzero post-offer failure probability, which we consider the next stage of this work. Our preliminary results in this latter suggest that post-offer failure probability significantly hurts the performance of our benchmark policy, pairs could be matched through unlimited length cycles; which is somehow not surprising due to the increasing risk in the post-offer failure. Lastly, it would be very interesting to identify and study key features in the design of kidney exchange policies when pairs are registered in multiple exchange programs; this latter models could model the decision of the pairs that could have more than one offer -e.g the one that offers first to decide accepting or rejecting an offer.

Chapter 4

Online scheduling for energy minimization with a constrained adversary

4.1 Introduction

We study an online version of the problem of scheduling a set of jobs with release times and deadlines on a single machine in order to minimize the total number of idle periods. We also study an online version of the problem of scheduling a set of jobs with release times and deadlines on a single machine in order to minimize the total energy consumed under a given machine energy consumption model. The offline problem of scheduling a set of jobs with release times and deadlines on a single machine, in order to minimize the total number of idle periods, was shown to be polynomial time solvable by Baptiste in [11]. When the switching energy costs are small enough to make a transition from high power state to low power state during any idle period less costly than staying in high power state while not processing, minimizing the total number of idle periods is equivalent to minimizing the total energy consumption. When switching costs are larger, the problem is more challenging. For which case, Baptiste, Chrobak and Durr in [12] provided a polynomial time algorithm in order to minimize the total energy consumption.

In many real-life problems, the existence of jobs and their characteristics are not known until they

are released, hence its online version is of special interest. Unfortunately, any online algorithm needs to process jobs as soon as they can be processed in order to guarantee that deadlines will be met. Previous work has focused on how to manage the energy consumption in idle periods [22; 23; 9].

In many situations, standard worst-case analysis (an offline adversary) may be pessimistic. For example, we may know that our system is unlikely to be so heavily loaded that jobs must run as soon as possible. To model this, we introduce a *constrained adversary*, who is limited in how much load may be placed on the machine. The *constrained adversary* is inspired in the adversarial model used in adversarial queuing theory. A constrained adversary is an arbitrary that send jobs under a restriction analogous to the load condition in queuing theory [14]. In this chapter, we consider the problem of minimizing the total energy consumed by a two power state device, to process jobs that are sent over time by a *constrained adversary*. Jobs can be preempted and deadlines need to be met. In this problem, an algorithm must decide when to schedule the jobs, as well as at which power state it should run at any time. The constrained adversary is defined by two parameters: $s_{\max}$ and $\rho_{\max}$. The former is the upper bound on the span of the jobs released by the constrained adversary an the latter is an upper bound on $\sum_{\{j:r_j \leq t < d_j\}} \frac{\omega_j}{d_j - r_j}$, in which ω_i, r_i and d_i are the workload, released time, deadline of a job i released by the constrained adversary and t is an arbitrary time.

The problem energy($\rho_{\max}$,$s_{\max}$) is the problem of minimizing the total energy consumed by a single machine with two power states to process an instance produced by a constrained adversary with parameters $\rho_{\max} \leq 1$ and $s_{\max}$. We provide an online algorithm to minimize the energy consumption for arbitrary values of power consumption rates. We also provide an online algorithm to minimize the total number of idle periods in the resulting schedule. In both cases, we provide upper bounds on the competitive ratio of our algorithms, and lower bounds on all online algorithms. The bounds presented hold for arbitrary power consumption rates of the processor.

Now we proceed to describe the organization of this chapter. As previously mentioned the differences in the energy consumption between schedules arises from two main features: the number and length of the idle periods and the sequence of power states followed by the algorithm during the idle periods. In Section 4.2 we address the first of these aspects, the number and length of the idle periods in the resulting schedule. We present our online algorithm LAZY, which is used by our energy saving algorithm DEFER to schedule the jobs. In Section 4.3 we consider

the special case of a processor's power management mechanism in which the processor always transits to zero energy consumption power state when there is no workload pending to be processed. Additionally the processor requires a non zero amount of energy to restart processing. Under this power management mechanism, minimizing the energy consumption is equivalent to minimizing the number of idle periods. We give an upper bound on its competitive ratio and the smaller lower bound on the competitive ratio of any online algorithm when the performance measure is the number of idle periods in the resulting schedule of our algorithm LAZY. Next in Section 4.4 we present our algorithm DEFER to solve problem energy($\rho_{\max}$,$s_{\max}$) and its analysis. DEFER schedule jobs according to the schedule produce by LAZY, and uses a competitive optimal deterministic algorithm, to decide the processor's sequence of power states and the elapsed time in each state. An asymptotic upper bound on its competitive ratio, and an asymptotic lower bound on the competitive ratio of any online algorithm are given. Finally in Section 4.5 we discuss open directions and conclusions.

Preliminaries All parameters in this chapter are non negative integers, unless otherwise noted. We assume that time is discrete. More specifically, it is divided into unitary length time intervals, $(t, t+1]$, where t is an integer. A job J_i is a 3-tuple (ω_i, r_i, d_i), with workload ω_i, release time r_i, and deadline d_i. The span of job J_i, $s_i = d_i - r_i$, is the difference between its deadline and its release time. The *density* of an arbitrary job J_i is $\rho_i = \frac{\omega_i}{s_i}$. There are n jobs, $J_1, \ldots, J_n$, indexed in order of their release date, which arrive over time. Until a job is released we are not aware of its parameters, but once it is released we know them.

The processor is a single machine with two power states, high and low. H and L are the energy consumed per unit of time processing in high power state and low power state respectively. We assume that U is the energy necessary to power up the processor and that the energy necessary to power down the processor is 0. This assumption can be made even if there is a cost to power-down, because every power-up is preceded by a power-down, thus, the total cost require to complete a transition can be charged when it power-up. The cost of the last power-down, when all the instance have been processed, it is incurred by any algorithm. We also assume that transitions among power states occur in a negligible fixed amount of time. We assume that at time zero the machine is in high power state and that when the last job is released, the processor is notified that it is the last

job of the instance. Given that at time 0 we assumed that the machine is in high power state, it follows that any algorithm that does not transit immediately to low power state at time 0 will have an unbounded competitive ratio, because the adversary could release no job and then the online algorithm will consume a positive amount of energy and the optimal offline algorithm will consume no energy.

In a *feasible schedule* all jobs meet their deadlines. For a given schedule, $\omega_i(t)$ is the amount of work of job J_i that has been processed by time t. The *set of active jobs* at time t, $A(t)$, is the set of indexes of jobs with release time at most t and deadline strictly greater than t which are not fully processed by time t, $A(t) = \{i : r_i \leq t < d_i, \omega_i(t) < \omega_i\}$. The set of *partially active jobs* at time t, $\bar{A}(t)$, is the set of indexes of jobs with release time at most t and deadline strictly greater than t which are totally processed by time t, $\bar{A}(t) = \{i \notin A(t) : r_i < t < d_i\}$. For every i in $A(t)$, the *active workload of job* J_i at time t, $Q(t, i)$, is the total workload of jobs that are in $A(t)$, and have deadline at most d_i, $Q(t, i) = \sum_{\{j \in A(t): d_j \leq d_i\}} \omega_j$. The *density of an instance* at time t, $\rho(t)$, is the sum of the density of all jobs that were release before time t and have deadline strictly larger than time t, $\sum_{\{i: r_i \leq t < d_i\}} \rho_i$. We assume that the jobs in an instance are released by a *constrained adversary*. A *constrained adversary* with parameters $\rho_{\max} \leq 1$ and $s_{\max}$ is an adversary such that for every time t and for every instance I that it produces, $\rho(t) \leq \rho_{\max}$, and all jobs in I have span at most $s_{\max}$.

In summary, in this chapter we design an online algorithm to schedule the jobs released by a constrained adversary in order to minimize the total energy required to process the jobs and meet their deadlines. We present an upper and a lower bound on the competitive ratio of our algorithm and on an arbitrary online algorithm respectively.

4.2 Online scheduling algorithm

In this section we present our online scheduling algorithm LAZY. From now onwards we will denote by `LAZY`(t) the algorithm used by LAZY at time t. The key idea of `LAZY`(t) relies on the following observation. By waiting as long as possible before starting to process, there will be a larger amount of workload pending to be processed, and more jobs could be processed without creating additional idle periods. Thus, there tend to be fewer and longer idle periods, which is less costly than more

frequent smaller idle periods.

4.2.1 The algorithm

4.2.1.1 The optimization problem

We first introduce an optimization problem used by LAZY(t). Given a partial schedule of an arbitrary instance of problem energy($\rho_{\max}$,$s_{\max}$) let t be the current time. We can assume that t is an integer because all jobs have integer release time and deadline, which implies that changes in the workload pending to be processed and in the density available to be used by the constrained adversary only occurs at integer time, unless otherwise specified. Recall that a feasible schedule completes all jobs by their deadlines.

Let us also assume without loss of generality that starting to process at time t the workload that is pending to be processed at time t we guarantee that all deadlines are going to be met. We should also assume that by time t all deadlines in the given partial schedule have been met. The problem that we describe in this section computes the latest time at which we could start processing workload pending to be processed at time t and still meet the deadline of those jobs and the deadlines of any future arrivals. We provide a dynamic program that solves this optimization problem when jobs have unit workload, no two jobs have the same deadline and no two jobs have the same release time. For the general case we do not know how to compute an exact solution to this optimization program.

Let u_1 be an arbitrary time in time interval $\left(t, \min_{j \in A(t)} d_j\right]$ and let u_2 be an arbitrary integer time such that $u_2 \geq u_1$. Define $S(A(t))$ as the set of feasible schedules for jobs in $A(t)$ in which no job is started to be processed before time u_1. For every S in $S(A(t))$ let $q(S)$ be the total processing time in $[u_1, u_2]$ if $S(A(t))$ is not empty, otherwise let $q(S) = \infty$. Then define $\upsilon(t, u_1, u_2)$ as the optimal value of:

$$\upsilon(t, u_1, u_2) \quad = \quad \min_{S \in S(A(t))} q(S),$$

that is the smallest amount of workload that an arbitrary feasible schedule $S \in S(A(t))$ processes before time u_2. Additionally define $\omega^{\mathrm{N}}(t, u_1, u_2)$ as the largest amount of workload that could be released by the adversary in time interval $[u_1, u_2)$ and need to be processed before time u_2 to guarantee that no deadline will be violated. Let $J = \{J_1, \ldots, J_k\}$ be the set the of jobs with release

time at most t. Let $J' = \{j_{k+1}, \ldots, j_h\}$ be a set of jobs with release time at least u_1, $u_1 > t$, such that the set of jobs $J \cup J'$ satisfies the adversary density constraint at any time. The adversary density constraint is defined as follows:

$$\sum_{i \in A(t)} \rho_i \quad \leq \quad s_{\max}.$$

We define define $S(J')$ as the set of feasible schedules for J'. For $S \in S(J')$ let $p(S)$ be the total processing time in $[u_1, u_2)$. Then $\omega^{\mathrm{N}}(t, u_1, u_2)$ is defined as:

$$\omega^{\mathrm{N}}(t, u_1, u_2) \quad = \quad \max_{J'} \min_{S \in S(J')} p(S).$$

We will refer as J'^* to an arbitrary element of $\arg\max_{J'} \min_{S \in S(J')} p(S)$. Notice that $\omega^{\mathrm{N}}(t, u_1, u_2)$ does only depend on the density of the jobs in J and is independent of the scheduling decisions.

We wish to compute the latest time u_1 for which for all $u_2 \geq u_1$ there exist a feasible schedule of jobs in $A(t) \cup J'^*$ with starting time at least u_1. In other words, we aim to compute the largest u_1 for which for all $u_2 \geq u_1$ workload in $A(t) \cup J'^*$ could be processed and its deadlines will be met. In Section 4.2.1.1 we present an optimization problem to find u_1 when the job's processing time is allowed to not be an integer and in Section 4.2.1.1 we present an optimization problem to find u_1 when the job's processing time is constrained to be integer. From now onwards we will refer to the constraint on the integrality of the workload - the workload is assumed to be an integer or not, as the workload's integrality constraint.

Non integer workload Let us consider the following optimization problem with decision variables u_1 and u_2.

Problem 1.

$$\begin{aligned}
&\underset{u_2}{\textit{minimize}}\ \underset{u_1}{\textit{maximize}} && u_1 && \\
&\textit{subject to} && u_1 + \omega^{\mathrm{N}}(t, u_1, u_2) + \upsilon(t, u_1, u_2) \leq u_2 && (4.2.1)\\
& && t < u_1 \leq \min_{j \in A(t)} d_i - \upsilon\left(t, t, \min_{j \in A(t)} d_i\right) && (4.2.2)\\
& && u_1 \leq u_2. && (4.2.3)
\end{aligned}$$

The inequality (4.2.1) is a necessary condition for the the existence of a feasible schedule with starting time at least u_1 of jobs in $A(t) \cup J'^*$, the inequality (4.2.2) implies that time u_1 is in the future of time t and it is smaller that the latest time at which we could start processing the workload in $A(t)$ to meet their deadlines, and inequality (4.2.3) implies that u_2 is always after u_1. Given that $\omega^{\mathrm{N}}(t, u_1, u_2) \geq 0$ and $\upsilon(t, u_1, u_2) \geq 0$, then it follows that inequality (4.2.3) is dominated by inequality (4.2.1) and it is redundant. Problem 1 computes for each value u_2 larger than u_1, the largest u_1 for which the necessary condition for the existence of a feasible schedule of jobs in the corresponding $A(t) \cup J'^*$ is guaranteed. Denote such value of u_1 as $u_1(u_2)$. Problem 1 returns $\min_{u_2 > t} u_1(u_2)$.

We describe next how to solve Problem 1. Before doing so, we establish some useful definitions. Let $U(t) = \{u_2 \in \mathbb{Z}^+ : u_2 > t\}$. Let us define the unidimensional function $f(u_2)$ for $u_2 \in U(t)$ as:

Problem 2.

$$
\begin{aligned}
f(u_2) = \underset{u_1}{\textit{maximize}} \quad & u_1 \\
\textit{subject to} \quad & u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, u_2) + \upsilon(t, u_1, u_2) \leq u_2 && (4.2.4) \\
& t \leq u_1 \leq \min_{j \in A(t)} d_i - \upsilon\left(t, t, \min_{j \in A(t)} d_i\right). && (4.2.5)
\end{aligned}
$$

In order to solve Problem 1 we first argue that there is an optimal solution u_2^* to problem:

$$\min_{u_2 \in U(t)} f(u_2)$$

in a finite subset of $U' \subset U$, with size at most $s_{\max}$. This fact together with definition of the feasible set of Problem 2 imply that there is an optimal solution to Problem 1 in the finite set

$$T = \left\{(u_1, u_2) : u_1 \leq u_2, u_2 \in U'\right\}.$$

The size of T just defined is upper bounded by $s_{\max}^2$.

We argue in Proposition 4.2.1 that there exists a J'^* such that all jobs in J'^* have deadline at most u_2 (release times of jobs in J'^* are at least u_1 by its definition). This result implies that to compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ suffices to consider a set of jobs with release time and deadline in $[u_1, u_2]$.

Proposition 4.2.1. *If the workload integrality is relaxed, for every* $u_1 \in \left[t, \min_{j \in A(t)} d_i - \upsilon\left(t, t, \min_{j \in A(t)} d_i\right)\right)$ *and* $u_2 \geq u_1$ *there exist an optimal solution* J'^* *to problem,*

$$\max_{J'} \min_{S \in S(J')} p(S)$$

such that all jobs in J'^* *have deadline at most* u_2*.*

Proof. Let J'_1 be an optimal solution to problem $\max_{J'} \min_{S \in S(J')} p(S)$ with the smallest total processing time. We may assume that at least one job in J_1 has deadline strictly larger than u_2 otherwise the proposition follows trivially. To prove this statement we will argue that all jobs in J'_1 have deadline at most u_2. To do so, we will show that if this is not the case then J'_1 is not an optimal solution to problem $\max_{J'} \min_{S \in S(J')} p(S)$ with the smallest total processing time, which is a contradiction.

Let $u = \max_{i \in S_0} d_i$. By assumption $u > u_2$. Let $J^u = J'_1$. From the set J^u define the following sets of jobs:

$$\begin{aligned} S_1^u &= \{j \in J^u : d_j = u, r_j \leq u - 2\} \\ S_2^u &= \{j \in J^u : d_j < u\} \cup \{j \in J^u : d_j = u, r_j = u - 1\} \\ S_3^u &= \emptyset. \end{aligned}$$

For each $v = u_2, \ldots, u-1$ construct an augmented instance J^v and sets S_1^v, S_2^v and S_3^v as follows:

1. Let $S_3^v = S_3^{v+1}$ and add the jobs in S_3^v to J^v.
2. For each job j_k in S_1^{v+1}:
   - Add to J^v a job with release time r_k, deadline $d_k - 1$ and workload $\frac{\omega_k}{d_k - r_k} \cdot (d_k - 1 - r_k)$. If $r_k \leq v - 2$ add this job to set S_1^v otherwise add it to the set S_2^v.
   - Add to J^v a job with release time $d_k - 1$, deadline d_k, and workload $\frac{\omega_k}{d_k - r_k}$. Add this new job to set S_3^v.
3. For each job J_k in S_2^{v+1} add the corresponding job to J^v. Additionally, if the job has deadline v and release time at most $v-2$ add it to the set S_1^v. Otherwise add it to the set S_2^v if deadline is strictly less than v, otherwise add it to S_3^v if deadline is $v + 1$.

In Figure 4.2.1 we illustrate the construction of set of jobs J^h, for $h = u_2, \ldots, u$ through an example.

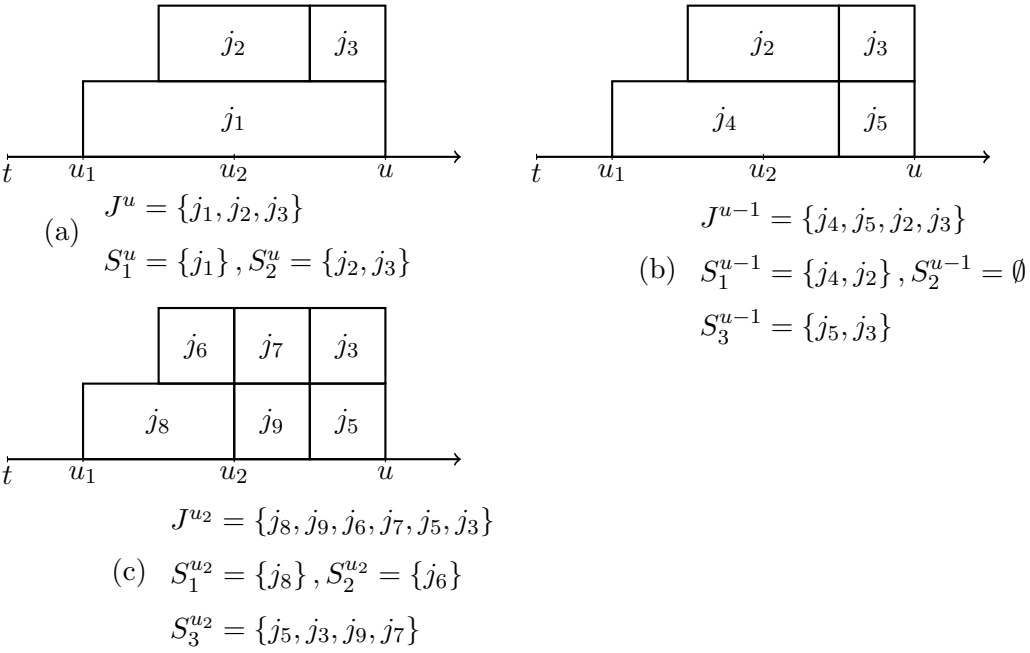


Figure 4.2.1: (a) Set of jobs J^u (b) Set of jobs J^{u-1} (c) Set of jobs J^{u_2}.

From the construction of J^{u-1} from $J^u = J'_1$ it follows that from each job k in S_1^u , two jobs in J^{u-1} are created, one with release time r_k, deadline $d_k - 1$ and workload $\frac{\omega_k}{d_k - r_k} \cdot (d_k - 1 - r_k)$, denote as job k_1, and another one with release time $d_k - 1$, deadline d_k and workload $\frac{\omega_k}{d_k - r_k}$, denote as job k_2. The total processing time of J^u is equal to the total processing time of J^{u-1} however the total amount of workload in J^{u-1} with deadline $u - 1$ is strictly larger than the one in J^u with deadline $u - 1$, which together with the construction of J^{u-1} from J^u and the adversary density constraint, imply that the total workload processed before $u - 1$ in a schedule of J^u that minimize the processing time before $u - 1$ is at most the total processing time before u in a schedule of J^{u-1} that minimize the processing time before $u - 1$. This fact together with the fact that the subset of jobs in J^{u-1} with deadline at most $u - 1$ has strictly less total processing time than J^u allow us to conclude that J'_1 is no the optimal solution to $\max_{J'} \min_{S \in S(J')} p(S)$ with the smallest total processing time. Because $\left\{ j \in J^{u-1} : d_j \leq u \right\}$ is also an optimal solution and it has strictly less total processing time.

□

A direct consequence of Proposition 4.2.1 is that to compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ we only need to consider the set of jobs that could be released by the constrained adversary which all jobs have release time in $u_1, \ldots u_2$ and deadline at most u_2. Before presenting our next result, we introduce a new definition. Define $\rho_{\text{apriori}}(t, u)$ for an arbitrary $u \geq t$ as:

$$\rho_{\text{apriori}}(t, u) = \rho_{\max} - \sum_{\left\{j \in A(t) \cup \bar{A}(t) : d_j > u\right\}} \rho_j.$$

Intuitively $\rho_{\text{apriori}}(t, u)$ is the density that is available to the adversary at time u to release new jobs if no release occurs from current time t up to time u. In the example in Figure 4.2.2 jobs do not necessarily have integer workload however release times and deadlines are integer. t time 0 job 1, $J_1 = (0, 10, 3)$ is released. Such job is fully processed by time 3.7, otherwise at time 1 a job with deadline 10 and workload 6.3 could be released and both jobs could not be processed on time. Consider that current time is 4, due to the adversary density constraint jobs 2 and 3, $J_2 = (7, 14, 3), J_3 = (10, 13, 1)$ could be released by the adversary after time 4, so together they constituted an augmented instance at time 4 but also on their own. However $J_4 = (5, 6, 1)$ cannot be released by the adversary due to the adversary density constraint, so it is not contain in any augmented instance at time 4. Notice that the area of the rectangle associated with each job corresponds to the workload of the job.

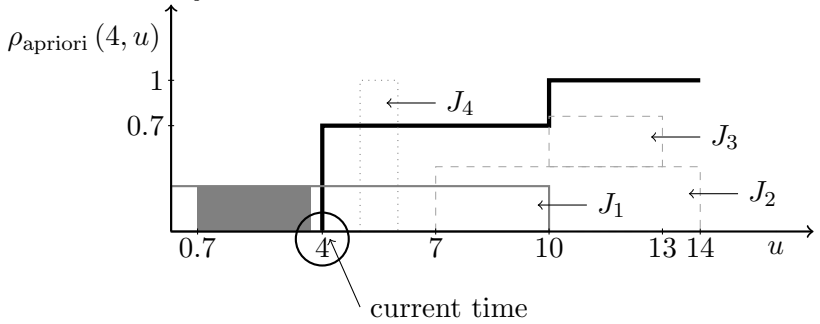


Figure 4.2.2: Example to illustrate the computation of $\omega^{\mathrm{N}}(t, u_1, u_2)$ when workload is allowed to not be integral

Now we show that $\omega^{\mathrm{N}}(t, u_1, u_2)$ is equal to the area under the curve defined by $\rho_{\text{apriori}}(t, u)$ for $u \in [u_1, u_2]$.

Proposition 4.2.2. *If t is the current time and $u_1 \in \left[t+1,\ \min_{j\in A(t)} d_i - \upsilon\left(t, t,\ \min_{j\in A(t)} d_i\right)\right]$, for any $u_2 \geq u_1$ it holds that,*

$$\omega^{\mathrm{N}}(t, u_1, u_2) \quad = \quad \sum_{h=u_1}^{u_2} \rho_{\text{apriori}}(t, h), \tag{4.2.6}$$

Proof. To prove this proposition we consider the following optimization problem in which there is one decision variable per possible release time-deadline combination that lies in time interval $[u_1, u_2]$. Decision variable $I_{t_i t_j}$ represent the workload of a job with deadline t_j and release time t_i.

Problem 3.

$$\begin{aligned} \omega^{\mathrm{N}}(t, u_1, u_2) = \underset{I_{ij}}{\textit{maximize}} \quad & \sum_{j\geq i+1}^{u_2} \sum_{i=\lceil u_1 \rceil}^{u_2-1} I_{ij} \\ \textit{subject to} \quad & \textstyle\sum_{j=\lceil u_1 \rceil}^{l} \sum_{k=l+1}^{u_2} \frac{I_{jk}}{k-j} \leq \rho_{\text{apriori}}(t, l), \quad \forall l = \lceil u_1 \rceil, \ldots u_2 - 1 \\ & I_{ij} \geq 0, i = \lceil u_1 \rceil, \ldots, u_2 - 1, j \ \text{ and } \ u_1 + 1, \ldots, u_2 \end{aligned} \tag{4.2.7}$$

The set of constraints defined by inequality (4.2.7) guarantee that the adversary density constraint is respected at any time l in time interval $[u_1, u_2)$. Intuitively the optimal solution to Problem 3 is given by the area under the curve $\rho_{\text{apriori}}(t, u)$ for $u \in [\lceil u_1 \rceil, u_2]$. Let us consider the dual of Problem 3:

Problem 4.

$$\begin{aligned} \omega^{\mathrm{N}}(t, u_1, u_2) = \underset{y_l}{\textit{minimize}} \quad & \sum_{l=\lceil u_1 \rceil}^{u_2-1} \rho_{\text{apriori}}(t, l) \cdot y_l \\ \textit{subject to} \quad & \textstyle\frac{\sum_{j=k}^{l} y_j}{l-k+1} \geq 1, \quad \forall l \geq k, l = \lceil u_1 \rceil, \ldots u_2 - 1, k = \lceil u_1 \rceil, \ldots, u_2 - 1 \\ & y_j \geq 0, j = \lceil u_1 \rceil, \ldots, u_2 - 1. \end{aligned} \tag{4.2.8}$$

The solution $y_j^* = 1$ for $j = \lceil u_1 \rceil, \ldots, u_2$ is a feasible solution to Problem 4 with objective value $\sum_{i=\lceil u_1 \rceil}^{u_2-1} \rho_{\text{apriori}}(t, i)$. Additionally the solution $I^*_{i(i+1)} = \rho_{\text{apriori}}(t, i)$ for $i = \lceil u_1 \rceil, u_2 - 1$ and 0 for all other decision variables, is a feasible solution to Problem 3 with objective value $\sum_{i=\lceil u_1 \rceil}^{u_2-1} \rho_{\text{apriori}}(t, i)$. From duality theory it follows that such solutions are optimal for their respective problems and so $\omega^{\mathrm{N}}(t, \lceil u_1 \rceil, u_2) = \omega^{\mathrm{N}}(t, \lceil u_1 \rceil, u_2) = \sum_{i=\lceil u_1 \rceil}^{u_2-1} \rho_{\text{apriori}}(t, i)$. □

Now we proceed to argue that to solve Problem 1 we just need to solve Problem 2 for a finite number of values of u_2, at most $|A(t)| \leq s_{\max}$ of them. Specifically we will argue that the optimal solution to Problem 2 is given by:

$$\min_{\{d_i : i \in A(t)\}} f(d_i).$$

We will first introduce some useful notation. Let S_p be a schedule of jobs in $A(t)$ such that for an arbitrary time u the total workload processed before time u is $\upsilon(t, t, u)$. Equivalently, S_p is a schedule of $A(t)$ in which jobs are processed as late as possible. Denote as $I(S_p)$ the set of bounded maximal time intervals in schedule S_p in which no workload unit is processed during the time interval and let us denote as $B(S_p)$ the set of maximal time intervals in schedule S_p in which workload is processed during all the time interval. For an arbitrary $I = [a, b) \in I(S_p)$, a is an integer number and b is a real number. Similarly, for an arbitrary $I = [b, a) \in B(S_p)$, b is a real number and a is an integer, see Figure 4.2.3 for an example. Assume that for an arbitrary $I = [a, b) \in I(S_p)$ a is not an integer. Given that all deadlines are integral it follows that the jobs that are processed in time interval $[\lfloor a \rfloor, a]$ in S_p have deadline at least $\lceil a \rceil$. Consider the schedule S'_p equal to S_p but it processes workload processed in S_p in interval $[\lfloor a \rfloor, a]$, $\lceil a \rceil - a$ units later. S'_p is a feasible schedule in which the workload processed before time a is smaller than $\upsilon(t, t, a)$, which contradicts definition of $\upsilon(t, t, a)$.

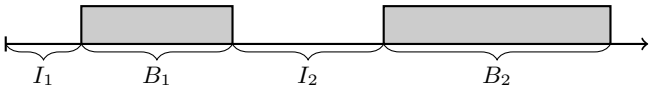


Figure 4.2.3: S_p is a schedule such that $I(S_p) = \{I_1, I_2\}$ and $S(S_p) = \{B_1, B_2\}$

Now we are ready to show that $f(u_2)$ is non decreasing on $\cup_{I \in I(S_p)} (I \cap \mathbb{Z})$ and non increasing on $\cup_{I \in B(S_p)} (I \cap \mathbb{Z})$, which imply that $f(u_2)$ is minimized at the supremum of some $I \in B(S_p)$. These facts together with the fact that $\{\sup I : I \in B(S_p)\} \subseteq \{d_i : i \in A(t)\}$ imply that the optimal value to Problem 1 is $\min_{\{d_i : i \in A(t)\}} f(d_i)$, which is the desired result.

Proposition 4.2.3. *$f(u_2)$ is non decreasing on $\bigcup\limits_{I \in I(S_p)} (I \cap \mathbb{Z})$.*

Proof. Let $I = [a, b)$ be an arbitrary interval in $I(S_p)$. Let u_1 be an arbitrary time in $[t, \min_{j \in A(t)} d_j - \upsilon\left(t, t, \min_{j \in A(t)} d_j\right)]$ From definition of set $I(S_p)$ it follows that $\upsilon(t, u_1, a + \delta) = \upsilon(t, u_1, a)$ for all

$\delta = 0, \ldots, \lfloor b \rfloor - a$. Then it follows that inequality (4.2.4) for $f(a+\delta)$ for an arbitrary $\delta = 0, \ldots \lfloor b \rfloor - a$ can be written as:

$$u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, a) + \sum_{j=a}^{a+\delta-1} \rho_{\text{apriori}}(t, j) + \upsilon(t, u_1, a) \quad \leq \quad a + \delta,$$

which can be rewritten as:

$$u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, a) + \upsilon(t, u_1, a) \quad \leq \quad a + \delta - \sum_{j=a}^{a+\delta-1} \rho_{\text{apriori}}(t, j).$$

Given that $\delta - \sum_{j=a}^{a+\delta-1} \rho_{\text{apriori}}(t, j)$ is increasing in δ, then it follows that $f(a+\delta) \geq f(a)$ for an arbitrary $\delta = 0, \ldots, \lfloor b \rfloor - a$, which is the desired result. □

Proposition 4.2.4. *$f(u_2)$ is non increasing on $\bigcup_{I \in B(S_p)} (I \cap \mathbb{Z})$.*

Proof. Let $I = [b, a)$ be an arbitrary interval in $B(S_p)$. Let u_1 be an arbitrary time in $[t, \min_{j \in A(t)} d_j - \upsilon\left(t, t, \min_{j \in A(t)} d_j\right)]$. From definition of set $B(S_p)$ it follows that $\upsilon(t, u_1, \lceil b \rceil + \delta) = \upsilon(t, u_1, \lceil b \rceil) + \delta$ for all $\delta = 0, \ldots, a - \lceil b \rceil$. Then it follows that inequality (4.2.4) of $f(\lceil b \rceil + \delta)$ for an arbitrary $\delta = 0, \ldots, a - \lceil b \rceil$ can be written as:

$$u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, \lceil b \rceil) + \sum_{j=\lceil b \rceil}^{\lceil b \rceil + \delta - 1} \rho_{\text{apriori}}(t, j) + \upsilon(t, u_1, \lceil b \rceil) + \delta \quad \leq \quad \lceil b \rceil + \delta,$$

which can be rewritten as:

$$u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, \lceil b \rceil) + \upsilon(t, u_1, \lceil b \rceil) \leq \lceil b \rceil - \sum_{j=\lceil b \rceil}^{\lceil b \rceil + \delta - 1} \rho_{\text{apriori}}(t, j).$$

Given that $-\sum_{j=\lceil b \rceil}^{\lceil b \rceil + \delta - 1} \rho_{\text{apriori}}(t, j)$ is decreasing in δ then it follows that $f(\lceil b \rceil) \geq f(\lceil b \rceil + \delta)$ for an arbitrary $\delta \in [0, a - \lceil b \rceil]$. □

In conclusion, when workload's integrality is relaxed, to solve optimization Problem 1 it suffices to compute 1) schedule S_p, and 2) $\omega^{\mathrm{N}}(t, t, d_j)$ for all $j \in A(t)$.

We finish this section arguing that the optimization problem 1 captures the desired features- that starting to process strictly later than time $\min_{u_2 \geq t} f(u_2)$ the adversary could release a set of jobs J such that at least one job will not meet its deadline. To show this we will argue that for an

arbitrary given u_2 value, if $u_1^* + 1$ is the smallest u_1 value for which inequality (4.2.1) is violated, then for all $u_2 \geq u_1 \geq u_1^*$, inequality (4.2.1) is also violated. If this is not the case then, $f(u_2) > u_1^*$ which implies that Problem 1 is not capturing the desired result. From assumption:

$$(u_1^* + 1) + \omega^{\mathrm{N}}\left(t, u_1^* + 1, u_2\right) + \upsilon\left(t, u_1^* + 1, u_2\right) \quad > \quad u_2 \tag{4.2.9}$$

Let us consider $u_1 + \delta \leq u_2$ for some arbitrary $\delta > 1$:

$$\begin{aligned}
(u_1^* + \delta) + \omega^{\mathrm{N}}\left(t, u_1^* + \delta, u_2\right) + \upsilon\left(t, t, u_2\right) &= (u_1^* + \delta) + \Big[\omega^{\mathrm{N}}\left(t, u_1^* + 1, u_2\right) \\
&\quad - \sum_{u=u_1^*+1}^{u_1^*+\delta} \rho_{\text{apriori}}\left(t, u\right)\Big] + \upsilon\left(t, t, u_2\right) \\
&\geq (u_1^* + \delta) + \left[\omega^{\mathrm{N}}\left(t, u_1^* + 1, u_2\right) - \delta\right] + \upsilon\left(t, t, u_2\right) \\
&> u_2.
\end{aligned}$$

thus for any $u_1 > u_1^*$ inequality (4.2.1) is violated. The last inequality follows from assumption in Inequality (4.2.9). Assume that u_1 is in the interval defined by the two consecutive elements of set $T(t)$: $[a_o, a_{o+1}]$. If this is the case then inequality (4.2.4) can be written as:

Workload is integral We will start this section by arguing that when workload's integrality is required Problem 5 computes the largest time at which we could start processing the pending workload and meet all the deadlines as well as the deadlines of any job that could be released by the adversary.

Problem 5.

$$\begin{aligned}
&\underset{u_2:u_2\geq u_1}{\textit{minimize}}\ \underset{u_1}{\textit{maximize}} && u_1 \\
&\textit{subject to} && (u_1) + \omega^{\mathrm{N}}\left(t, u_1, u_2\right) + \upsilon\left(t, u_1, u_2\right) \leq u_2 && (4.2.10) \\
& && t < u_1 \leq \min_{j\in A(t)} d_i - \upsilon\left(t, t, \min_{j\in A(t)} d_i\right) && (4.2.11) \\
& && u_1 \leq u_2. \quad u_1, u_2 \in \mathbb{Z}^+ && (4.2.12)
\end{aligned}$$

To do so, we first argue that for a given u_2, $\omega^{\mathrm{N}}\left(t, u_1 + 1, u_2\right) \geq \omega^{\mathrm{N}}\left(t, u_1, u_2\right) - 1$ for any u_1 that satisfied Inequality (4.2.2).

Proposition 4.2.5. *If u_1 satisfies Inequality (4.2.2) and the constrained adversary only releases jobs with integral workload, then for an arbitrary $u_2 \geq u_1$ it holds that:*

$$\omega^{\mathrm{N}}(t, u_1 + 1, u_2) \quad \geq \quad \omega^{\mathrm{N}}(t, u_1, u_2) - 1.$$

Proof. For a given u_1, u_2 let J'^* be the optimal solution to $\max_{J'} \min S \in S(J')p(S)$ with the smallest largest deadline. Consider the set of jobs J' such that for every job j in J'^* there is a job with release time $r_j + 1$, deadline $d_j + 1$ and workload ω_j. From the fact that $\rho_{\text{apriori}}(t, u)$ is a non decreasing function in u it follows that J' is a set of jobs that could be released by the constrained adversary in which all jobs have release time at least $u_1 + 1$. From assumption that at least $\omega^{\mathrm{N}}(t, u_1, u_2)$ units of the total workload of jobs J'^* are processed in $[u_1, u_2]$ in any feasible schedule together with the construction of J' from J'^* then it follows that at least $\omega^{\mathrm{N}}(t, u_1, u_2)$ units of the total workload of jobs in J' are processed in $[u_1 + 1, u_2 + 1]$ in any feasible schedule, thus at least $\omega^{\mathrm{N}}(t, u_1, u_2) - 1$ workload units are processed in $[u_1 + 1, u_2]$ in any feasible schedule of jobs in J', which is the desired result. □

Proposition 4.2.6. *If u_1 satisfies Inequality (4.2.2) and $u_1 + \omega^{\mathrm{N}}(t, u_1, u_2) + \upsilon(t, t, u_2) > u_2$ then:*

$$u + \omega^{\mathrm{N}}(t, u, u_2) + \upsilon(t, t, u_2) \quad > \quad u_2 \quad \forall u \geq u_1$$

Proof. Let $u = u_1 + \delta$ for some arbitrary $\delta \in \mathbb{Z}^+$. We will prove this statement by induction on δ. For the base case let $\delta = 1$. From assumption it holds that:

$$u_1 + \omega^{\mathrm{N}}(t, u_1 + 1, u_2) + \upsilon(t, u_1 + 1, u_2) \quad > \quad u_2 \tag{4.2.13}$$

Let us consider $u = u_1 + 1$:

$$\begin{aligned}
(u_1^* + 1) + \omega^{\mathrm{N}}(t, u_1^* + 1, u_2) + \upsilon(t, t, u_2) \quad &\geq \quad (u_1^* + 1) + \left[\omega^{\mathrm{N}}(t, u_1^*, u_2) - 1\right] \\
&\quad\quad +\upsilon(t, t, u_2) \\
&\geq \quad u_1 + \omega^{\mathrm{N}}(t, u_1, u_2) + \upsilon(t, t, u_2) \\
&> \quad u_2,
\end{aligned}$$

the last inequality follows from Inequality 4.2.13. The inductive step follows the base case. □

We argue next that once workload's integrality is required, computing $\omega^{\mathrm{N}}(t, u_1, u_2)$ appears to become "difficult". Even though we do not present hardness results of this problem, we will described different features that makes computing $\omega^{\mathrm{N}}(t, u_1, u_2)$ difficult. Notice that to establish an efficient procedure to solve Problem 1 when workload's integrality is relaxed, it was critical to find a bounded subset of the feasible region in which we could guarantee lies an optimal solution, as well as to be able to efficiently compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ over this bounded subset of the feasible region. When the workload is required to be integer we no longer know how to compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ efficiently, we also don't know how to bound the feasible region for the corresponding optimization problem, Problem 5, in the general case.

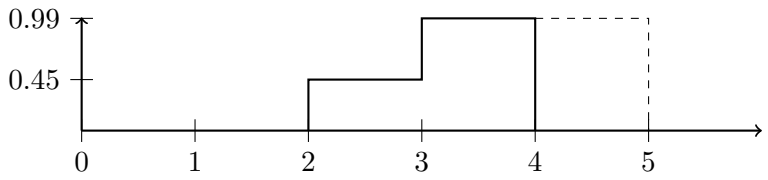
The example in Figure 4.2.4 shows that when the adversary could release jobs with integer workload, it is no longer true in general that there exist a set of jobs J'^{*}, $J'^{*} \in \arg\max_{J'} \min_{S \in S(J')} p(S)$, such that all job's deadlines are at most u_2. In this example, the adversary can not release any job with integral workload in $[2, 4]$ with deadline at most 4, however it can release an unit job at time 2 with deadline 5 and an unit job with deadline 5 at time 3, at least one of those jobs is processed before time 4. Given that the area under the density profile up to time 4, $\sum_{u=2}^{4} \rho_{\text{apriori}}(t, u) = 1.44$, it follows that $\omega^{\mathrm{N}}(2, 2, 4) \leq 1$, thus the given set of jobs is $\arg\max_{J'} \min_{S \in S(J')} p(S)$.

We explain next why the procedure to solve Problem 1 used when workload is not necessarily integral fails to solve Problem 5. Consider the following optimization problem:

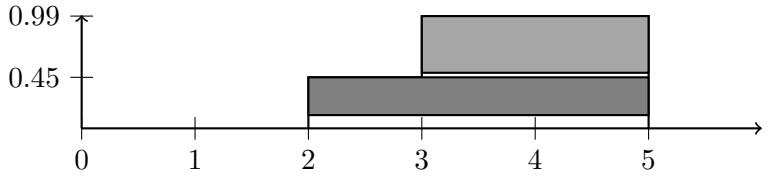
Problem 6.

$$
\begin{aligned}
g(d) = \underset{I_{ij}}{\textit{maximize}} \quad & \sum_{j \geq i+1}^{d} \sum_{i=u_1}^{d-1} I_{ij} \\
\textit{subject to} \quad & \sum_{j=u_1}^{l} \sum_{k=l+1}^{d} \frac{I_{jk}}{k-j} \leq \rho_{\text{apriori}}(t, l), \quad \forall l = u_1, \ldots d-1 \\
& I_{ij} \in \mathbb{Z}^{+} \cup \{0\}, i = u_1, \ldots, d-1, j \ \text{ and } \ u_1 + 1, \ldots, d
\end{aligned}
\tag{4.2.14}
$$

If we relax the workload's integrality constraint in Problem 6, equivalently $I_{ij} \in \mathbb{R}^{+} \cup \{0\}$, it follows that Problem 3 and Problem 6 are the same, and thus it follows that for arbitrary $d > u_1$, $g(d) = \omega^{\mathrm{N}}(t, u_1, d)$, which together with the closed form solution of $\omega^{\mathrm{N}}(t, u_1, u_2)$ in inequality 4.2.6 allow us to conclude that $g(d+1) \leq g(d) + 1$ when workload is not necessarily integral. This result is fundamental in computing $\omega^{\mathrm{N}}(t, u_1, u_2)$ for an arbitrary u_2, as well as in the proofs of Proposition 4.2.3 and Proposition 4.2.4 which allow us to fully characterize the set of local minimum



(a) No job with deadline at most 4 can be release after time 2.



(b) Two jobs with deadline 5 could be release after time 2 and at least one should be processed before time 4.

Figure 4.2.4: Example that shows that when workload needs to be integer it is no longer true that there exist a set of jobs J'^*, $J'^* \in \arg\max_{J'} \min_{S \in S(J')} p(S)$, such that all job's deadlines are at most u_2

of $f(u)$ with domain $\{u \in \mathbb{Z} : u \geq \lceil u_1 \rceil\}$, thus solving Problem 1 when the workload's integrality constraint is lifted. We proceed to address in detail these two problems.

About computing $\omega^{\mathrm{N}}(t, u_1, u_2)$, when workload's integrality is lifted we could prove that $g(d+1) \leq g(d) + 1$, thus $\omega^{\mathrm{N}}(t, u_1 + 1, u_2) \leq \omega^{\mathrm{N}}(t, u_1, u_2) + 1$. This result implies that $\omega^{\mathrm{N}}(t, u_1, u_2) = g(u_2)$ when workload's integrality constraint is lifted, which can be computed efficiently. When it does not hold then it is not difficult to see that $\omega^{\mathrm{N}}(t, u_1, u_2) = \max_{d \geq u_2} [g(d) - (d - u_2)]$ and unless $\omega^{\mathrm{N}}(t, u_1, u_2)$ reach its upper bound (floor of the area under the density profile $\rho_{\text{apriori}}(t, u)$ for $u_1 \leq u \leq u_2$) for some feasible value of d, we do not know how to provide a certificate of optimality for a given d value. Thus we do not know how to compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ when workload released by the constrained adversary is integral in the general case.

Suppose we could compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ when workload's integrality is required. It is no longer true that $\omega^{\mathrm{N}}(t, u_1, u_2) \leq \omega^{\mathrm{N}}(t, u_1, u_2)$ as shown in the example in Figure 4.2.4. It implies that $f(u_2)$ is no longer non decreasing for all $u_2 \in \bigcup_{I \in I(S_p)} (I \cap \mathbb{Z})$. Consider constraint 4.2.4 in definition of $f(u_2)$. For a given u_1, u_2, without been able to guarantee that $\omega^{\mathrm{N}}(t, u_1, u_2) \leq \omega^{\mathrm{N}}(t, u_1, u_2)$, u_1

may not be a feasible solution to $f(u_2+1)$ but it is a feasible solution to $f(u_2)$, which is equivalent to say that $f(u_2+1) < f(u_2)$ and thus $f(u)$is no longer non decreasing. The implication of it is that we no longer know how to find a finite countable set of values u_2 in which we guarantee there is an optimal solution to the problem $\min_{u_2 \geq u_1} f(u_2)$. Moreover we do not know how to provide a certificate of optimality for a given u_2 in the general case. We can conclude that for any set of instances for which we could guarantee that:

$$\omega^{\mathrm{N}}(t, u_1, u_2+1) \leq \omega^{\mathrm{N}}(t, u_1, u_2) + 1 \tag{4.2.15}$$

the procedure used to solve Problem 1 will solve Problem 5. Clearly, in the set of instances in which all jobs have unit workload, not two jobs have the same deadline Inequality (4.2.15) holds. Even under these simplifications, we do not know how to compute $\omega^{\mathrm{N}}(t, u_1, u_2)$ efficiently when workload is constrained to be integral. Next we show a dynamic programming approach and its running time analysis.

In Corollary 4.2.1 we argued that for a given u_1, u_2, $\omega^{\mathrm{N}}(t, u_1, u_2) = g(u_2)$ for the set of instances in which all jobs have unit workload and no two jobs have the same deadline.

Corollary 4.2.1. *If the adversary only releases jobs with unit workload and no two jobs have the same deadline then it follows that for a given u_1, u_2 there exist a*

$$J'^{*} \in \arg\max_{J'} \min_{S \in S(J')} p(S)$$

such that $\max_{j \in J'^{*}} d_j \leq u_2$.

Proof. Let us consider an augmented instance $J \in \arg\max_{J'} \min_{S \in S(J')} p(S)$ with the smallest number of jobs. Define d as $d = \max_{j \in J} d_i$. We may assume that $d > u_2$, otherwise the desired result follows trivially. Given that jobs have unit workload and that not two jobs can have the same deadline, it follows that the schedule in which every job j in J is processed in time interval $[d_j - 1, d_j]$ is a feasible schedule, and it is clearly an optimal solution to $\min_{S \in S(J)} p(S)$. It implies that there is a subset of jobs of J with deadline at most u_2 and total workload $\omega^{\mathrm{N}}(t, u_1, u_2)$, contradicting the assumption that J was the instance in $\max_{J'} \min_{S \in S(J')} p(S)$ with smallest number of jobs. □

Next we formulate Problem 3 as a dynamic program, which has running time $\Omega\left(\left(\frac{s_{\max}}{2}\right)!\right)$ and $O\left(s_{\max}^2\, s_{\max}!\right)$. Before stating the dynamic programming recursion to solve Problem 6 we introduce

some useful notation. Let $\vec{\rho}_{\text{apriori}}$ be a vector with $u_2 - u_1$ components in which the i^{th} component takes value $\rho_{\text{apriori}}(t, u_1 + i - 1)$. We consider the subproblem $v^{\text{N}}(t, u_1, d, \vec{w}_d)$ which includes the jobs with release time at least u_1, deadline at most d under density profile $\vec{w}_d$. The density profile $\vec{w}_d$ corresponds to a density profile from time u_1 to time d, thus it is a vector with $d-u_1$ dimensions. Let $\vec{\rho}_{jd}$ for some $j \in \{u_1, \ldots, k-1\}$ and some $j \leq d \leq u_2$ be a vector with dimension $d-u_1$ in which the last $d-j$ components have value $\frac{1}{d-j}$, and the rest have value zero. Let $I^{OPT}(\vec{w}_d)$ and $f_d(\vec{w}_d)$ denote an optimal solution and the optimal value for subproblem $v^{\text{N}}(t, u_1, d, \vec{w}_d)$. Notice that $I^{OPT}(\vec{w}_d)$ is a vector with $\frac{(d-u_1)(d-u_1+1)}{2}$ components. It has one component per release time-deadline pair that may occur in time interval $[u_1, d]$. We set $f_d(\vec{w}_d) = -\infty$ if the subproblem $v^{\text{N}}(t, u_1, d, \vec{w}_d)$ is infeasible. Notice that infeasibility of $f_d(\vec{w}_d)$ arises when at least one component of $\vec{w}_d$ takes value outside the interval $[0, \rho_{\max}]$. In an optimal solution to subproblem $v^{\text{N}}(t, u_1, d, \vec{w}_d)$ at most one variable among variables in $I^{OPT}(\vec{w}_d)$ that corresponds to the job with deadline d has value one and the rest have value zero. If the component of $I^{OPT}(d, \vec{w}_d)$ corresponding to a job with release time j and deadline d takes value one, then $f_d(\vec{w}_d) = f_{d-1}(\vec{w}_d[1:d-1] - \vec{\rho}_{jd}[1:d-1]) + 1$, the optimal value of subproblem $v^{\text{N}}(t, u_1, d-1, \vec{w}_d[1:d-1] - \vec{\rho}_{jd}[1:d-1])$ plus one (because all jobs have unit workload). If all components of $I^{OPT}(\vec{w}_d)$ take value zero, then $f_d(\vec{w}_d) = f_{d-1}(\vec{w}_d[1:d-1])$, the optimal value of the subproblem $v^{\text{N}}(t, u_1, d-1, \vec{w}_d[1:d-1])$. Taking the maximum of the preceding objective values gives the optimal solution of problem $v^{\text{N}}(t, u_1, d, \vec{w}_d)$. Denote $c_j(\vec{w}_d)$ for $j = u_1, \ldots, d-1$ as

$$c_j(\vec{w}_d) = \begin{cases} 1 \text{ if } w_{d-1} \geq \frac{1}{d-j}, \ldots, w_{d-j} \geq \frac{1}{d-j}, \\ 0 \text{ Otherwise,} \end{cases} \tag{4.2.16}$$

which takes value one if under the density profile $\vec{w}_d$ it is feasible to release a job with release time j and deadline d, and zero otherwise. Thus, the dynamic programming recursion can be stated as

$$f_d(\vec{w}_d) = \max_{j=u_1,\ldots,d-1} \left[f_{d-1}(\vec{w}_d[1:d-1] - \vec{\rho}_{jd}[1:d-1]) + c_j(\vec{w}_d) \right] \quad \text{for } d = u_1 + 1, \ldots, u_2, \tag{4.2.17}$$

with initial condition $f_{u_1}(\emptyset) = 0$. The objective is to compute $f_{u_2}(\rho_{\text{apriori}})$.

We proceed to analyze the running time of the dynamic program defined by the recursion in equation (4.2.17). To do so we will first argue that the number of subproblems $v^{\text{N}}(t, u_1, d, \vec{w}_d)$

needed to be solved to compute $f_{u_2}(\rho_{\text{apriori}})$ for a fix $d \in \{1, \ldots, u_2\}$ is

$$\sum_{k=0}^{\min\{u_2-d-1,d\}} \binom{u_2-d-1}{k} \binom{d}{k} k! \, (d+1-k) \,. \tag{4.2.18}$$

Notice that problems of type $\upsilon^{\text{N}}(t, u_1, d, \vec{w}_d)$ arise when solving problems $\upsilon^{\text{N}}(t, u_1, d+1, \vec{w}_{d+1})$, for some $d \in \{u_1, \ldots, u_2 - 1\}$. The number of different subproblems $\upsilon^{\text{N}}(t, u_1, d, \vec{w}_d)$ is determined by the number of different w_d profiles that may arise as well as by the number of jobs with deadline $d+1$ that should be considered under a given profile.

The different w_d profiles that $\upsilon^{\text{N}}(t, u_1, d, \vec{w}_d)$ could receive as parameter are a function of the jobs that could be released with deadline strictly larger than $d+1$ and release time at most d, see Figure 4.2.5. In Figure 4.2.5 we consider the subproblems with $w_d = \rho_{\text{apriori}} - \vec{\rho}_{(u_1+1)(d+2)} - \vec{\rho}_{0(d+1)}$, $w_d = \rho_{\text{apriori}} - \vec{\rho}_{(u_1+1)(d+2)} - \vec{\rho}_{1(d+1)}$ and $w_d = \rho_{\text{apriori}} - \vec{\rho}_{(u_1+1)(d+2)} - \vec{\rho}_{2(d+1)}$. Let us consider an

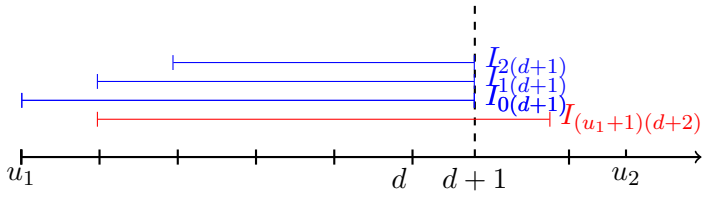


Figure 4.2.5: The problems $\upsilon^{\text{N}}(t, u_1, d, \vec{w}_d)$ arise while solving problems $\upsilon^{\text{N}}(t, u_1, d+1, \vec{w}_{d+1})$

arbitrary density profile $\vec{w}_{u_2}$ in which there are exactly k jobs with release time strictly smaller than d and deadline in $(d+1, u_2]$. There are at most $\binom{u_2-d-1}{k}$ sets of size k of different deadlines greater than $d+1$, $\binom{d-u_1}{k}$ set of size k of different release times with value at most $d-1$ and $k!$ different ways to match the k different deadlines with the k different release times. Notice that profiles with one job release at time d, appears among profiles in which $k-1$ jobs have release times at most $d-1$ and deadline in $(d+1, u_2]$, which explain why we only consider jobs with release time $1, \ldots, d-1$ when counting the different profiles that arise when there are exactly k jobs with release time strictly smaller than d and deadline in $(d+1, u_2]$. Another way to see why we will be double counting if we consider jobs with release time d is that a job release at time d does not affect $\vec{w}_d$. From the constraint that no two jobs are released at the same time and given that in the profile $\vec{w}_d$ there are k jobs with release time at most d and deadline greater than $d+1$, it follows that among the $d+1$ release times prior to d, there are $d+1-k-u_1$ available release times slots

to release jobs with deadline $d+1$. Considering that k ranges from 0 to $\min\{d-u_1, u_2-d-1\}$, thus the total number of different density profiles w_d that could arise is given by

$$\sum_{k=0}^{\min\{u_2-d-1,d-u_1\}} \binom{u_2-d-1}{k}\binom{d-u_1}{k} k!\,(d+1-k-u_1),$$

which corresponds to inequality (4.2.18). Then we can conclude that

$$\sum_{d=u_1+1}^{u_2-1}\sum_{k=0}^{\min\{u_2-d-1,d-u_1\}} \binom{u_2-d-1}{k}\binom{d-u_1}{k} k!\,(d+1-k-u_1)$$

problems need to be solved in order to compute $v^{\mathrm{N}}(t,u_1,u_2,\vec{w}_{u_2})$, equivalently the optimal value to Problem 6.

Now we proceed to prove the lower and upper bound in the running time. To do so, we consider the case in which u_2-u_1 is an even number and when it is an odd number. Assume that u_2-u_1 is an even number. The number of $v^{\mathrm{N}}\left(t,u_1,\frac{u_1+u_2}{2},\vec{w}_{\frac{u_1+u_2}{2}}\right)$ subproblems that are solved for $k=\frac{u_2-u_1}{2}-1$ it is a lower bound on the total number of subproblems that are solved to compute $v^{\mathrm{N}}(t,u_1,u_2,\vec{w}_{u_2})$ using the proposed dynamic program.

$$\begin{aligned}\sum_{d=u_1+1}^{u_2-1}\sum_{k=0}^{\min\{u_2-d-1,d-u_1\}} \binom{u_2-d-1}{k}\binom{d-u_1}{k} k!\,(d+1-k-u_1) \geq \quad & 2\left(\frac{u_2-u_1}{2}\right)\left(\frac{u_2-u_1}{2}-1\right)! \\ = \quad & 2\cdot\frac{u_2-u_1}{2}!. \end{aligned} \tag{4.2.19}$$

This together with the fact that $u_2-u_1\leq s_{\max}$, imply the desired result. The dynamic program defined by recursion in equation (4.2.17) has running time $\Omega\left(\frac{s_{max}}{2}!\right)$. The argument for the case in which u_2-u_1 is odd follows similarly. In this case we find a lower bound using the total number of subproblems $v^{\mathrm{N}}\left(t,u_1,\frac{u_1+u_2-1}{2},\vec{w}_{\frac{u_1+u_2-1}{2}}\right)$ that are solved for $k=\frac{u_2-u_1-1}{2}$.

Next, we will argue that the dynamic program defined by the recursion in equation (4.2.17) has worst case running time $O(s_{\max}!)$. We first argue that

$$\binom{u_2-d-1}{k}\binom{d-u_1}{k} k!\,(d+1-k-u_1) \leq (u_2-u_1)! \tag{4.2.20}$$

for some fix k and d. The left hand side of inequality (4.2.20) can be written as:

$$\begin{aligned}\binom{u_2-d-1}{k}\binom{d-u_1}{k} k!\,(d+1-k-u_1) &\leq \binom{(u_2-u_1)-(d+1-u_1)}{k}\frac{(d-u_1)!}{(d-u_1-k)!}(d-u_1-(k-1)) \\ &\leq \binom{(u_2-u_1)-(d+1-u_1)}{k}\frac{(d-u_1+1)!}{(d-u_1-k)!} \\ &\leq ((u_2-u_1)-(d+1-u_1))!\,(d+1-u_1)! \\ &= (u_2-u_1)!\,\frac{(d+1-u_1)!}{\prod_{i=0}^{d-u_1}(u_2-u_1-i)} \\ &= (u_2-u_1)!\prod_{i=0}^{d-u_1}\frac{(u_2-u_1)-(u_2-d-1+i)}{(u_2-u_1)-i} \end{aligned} \tag{4.2.21}$$

$$\leq \quad (u_2 - u_1)! \tag{4.2.22}$$

The inequality (4.2.22) follows from the fact that $\frac{(u_2-u_1)-(u_2-d-1+i)}{(u_2-u_1)-i} \leq 1$ for all $i \in \{0, \ldots, d-u_1\}$, because $u_2 - d - 1 + i \geq i$. We are left to argue that

$$\sum_{d=u_1+1}^{u_2-1} \sum_{k=0}^{\min\{u_2-d-1, d-u_1\}} 1 \leq (u_2 - u_1)^2.$$

It follows because

$$\begin{aligned}
\sum_{d=u_1+1}^{u_2-1} \sum_{k=0}^{\min\{u_2-d-1, d-u_1\}} 1 &= \sum_{d=u_1+1}^{u_2-1} \min\{u_2 - d - 1, d - u_1\} + 1 \\
&= \sum_{i=1}^{u_2-u_1} \min\{(u_2 - u_1 - 1) - i, i\} + 1 \\
&= \sum_{i=1}^{\frac{u_2-u_1-1}{2}} (u_2 - u_1 - i) + \sum_{i=\frac{u_2-u_1-1}{2}+1}^{u_2-u_1-1} i + 1 \\
&= \sum_{i=1}^{\frac{u_2-u_1-1}{2}} (u_2 - u_1 - i) + \sum j = 0^{\frac{u_2-u_1-1}{2}} u_2 - u_1 - j \\
&= (u_2 - u_1) + 2 \cdot \sum_{i=1}^{\frac{u_2-u_1-1}{2}} (u_2 - u_1 - i) \\
&= (u_2 - u_1) + 2 \cdot (u_2 - u_1) \cdot \left(\frac{u_2 - u_1 - 1}{2}\right) - \sum_{i=1}^{\frac{u_2-u_1-1}{2}-1} i \\
&= (u_2 - u_1)^2 - \frac{\left(\frac{u_2-u_1-1}{2} - 1\right)\left(\frac{u_2-u_1-1}{2}\right)}{2} \\
&\leq (u_2 - u_1)^2,
\end{aligned}$$

which implies that the total number of terms in the left hand side of inequality (4.2.19) is at most $(u_2-u_1)^2$. In conclusion, in a worst case analysis each term in the left hand side of inequality (4.2.19) is bounded above by $s_{\max}!$ and given that there are at most $s_{\max}^2$ terms and $u_2 - u_1 \leq s_{\max}$, it follows that the dynamic program defined by equation (4.2.17) has running time $O\left(s_{max}^2\, s_{\max}!\right)$.

We proceed to show that to solve Problem 5 we just need to compute $f(u_2)$ for $u_2 \in \{d_j : j \in A(t)\}$. The proofs of Proposition 4.2.7 and 4.2.8 mimic the proofs of Proposition 4.2.3 and 4.2.4 respectively.

Proposition 4.2.7. *If the constrained adversary is constrained to release only unit jobs and for which no two jobs have the same deadline,*

$$f(u_2+1) \quad \geq \quad f(u_2), \quad \forall u_2 \in \underset{I \in I(S_p)}{\cup} (I \cap \mathbb{Z}).$$

Proof. Let $I = [a, b)$ be an arbitrary interval in $I(S_p)$. Let u_2 be an arbitrary element of I. Assume for the sake of contradiction that $f(u_2+1) < f(u_2)$ for an arbitrary $u_2 \in I$. Let $f(u_2) = u_1^*$. It implies that u_1^* is not a feasible solution to $f(u_2+1)$:

$$u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2+1) + \upsilon(t, t, u_2+1) \quad > \quad u_2 + 1.$$

Given that $\omega^{\mathrm{N}}(t, u_1^*, u_2+1) \leq \omega^{\mathrm{N}}(t, u_1^*, u_2) + 1$ and that $\upsilon(t, t, u_2+1) = \upsilon(t, t, u_2)$ for all $u_2 \in I$ then it follows that:

$$\begin{aligned} u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2) + \upsilon(t, t, u_2) \quad &\geq \quad u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2+1) - 1 + \upsilon(t, t, u_2+1) \\ &> \quad u_2, \end{aligned}$$

which contradicts optimality of u_1^* to $f(u_2)$. □

Proposition 4.2.8. *If the constrained adversary is constrained to release only unit jobs and for which no two jobs have the same deadline,*

$$f(u_2+1) \quad \leq \quad f(u_2) \quad \forall u_2 \underset{I \in B(S_p)}{\cup} (I \cap \mathbb{Z}).$$

Proof. Let $I = [b, a)$ be an arbitrary interval in $B(S_p)$. Let u_2 be an arbitrary element of I. Assume for the sake of contradiction that $f(u_2+1) > f(u_2)$ for an arbitrary $u_2 \in I$. Let $f(u_2+1) = u_1^*$. It implies that u_1^* is not a feasible solution to $f(u_2)$:

$$u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2) + \upsilon(t, t, u_2) \quad > \quad u_2.$$

Given that $\omega^{\mathrm{N}}(t, u_1^*, u_2+1) \geq \omega^{\mathrm{N}}(t, u_1^*, u_2)$ and that $\upsilon(t, t, u_2+1) = \upsilon(t, t, u_2) + 1$ for all $u_2 \in I$ then it follows that:

$$\begin{aligned} u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2+1) + \upsilon(t, t, u_2+1) \quad &\geq \quad u_1^* + \omega^{\mathrm{N}}(t, u_1^*, u_2) + \upsilon(t, t, u_2+1) + 1 \\ &> \quad u_2 + 1, \end{aligned}$$

which contradicts optimality of u_1^* to $f(u_2)$.

□

The results in Proposition 4.2.7 and Proposition 4.2.8 allow us to conclude that there is an optimal solution to solve problem $\min_{u_2 \in \mathbb{Z}^+} f(u_2)$ in the set:

$$\{\inf\{I\} : I \in I(S_p)\},$$

for the corresponding S_p.

We just argued that to solve problem 2 we could only compute $\min_{u_2 \geq u_1} f(u_2)$ for a finite and well defined number of u_2 values. Given that $\{\inf\{I\} : I \in I(S_p)\} \subseteq \{d_i : i \in A(u_1)\}$, then it follows that:

$$\min_{u_2 \geq u_1} f(u_2) \quad = \quad \min_{i \in A(u_1)} f(d_i).$$

Then we can conclude that to solve optimization problem 1 for a given u_1 and the corresponding density profile, someone could evaluate function $f(u_2)$ for every $u_2 \in \{d_i : i \in A(u_1)\}$ and pick the smallest value.

4.2.1.2 START(t)

We gave a procedure START(t) which is used by LAZY(t) to compute when it should start processing the pending workload after an idle period. It implies that we assume that during time interval $[t-1, t)$ no workload unit is processed. START(t) receives as input the information regarding jobs with indexes in the set $A(t) \cup \bar{A}(t)$, that is the set of jobs that have been released by time t and have deadline strictly larger than t and compute a value which we will call $T_{\text{critic}}(t)$. In Section 4.2.2 we will argue that $T_{\text{critic}}(t)$ is the latest time at which LAZY(t) could start processing the workload that is pending to be processed at time t and guarantees that all deadlines will be met. Notice that this statement holds under the information known up to time t and if no job is released before time START(t), then START(t) is the latest time at which LAZY(t) could start processing the workload that is pending to be processed at time t and guarantees that all deadlines will be met.

```
START(t)
1  if A(t) ≠ ∅
2        for i ∈ A(t)
3              Compute min_{j∈A(t)} f(d_j)
4         if min_{j∈A(t)} f(d_j) = t
5              return min_{j∈A(t)} f(d_j)
6        else
7                    return START(min_{j∈A(t)} f(d_j))
8  else
9              return ∞
```

4.2.1.3 On the dynamics of the processor

We use *power(t)* and *status(t)* to describe the processor's state at time t. The variable *power*(t) can take values H, L and 0 and the variable *status*(t) can take values `process` and `not_process`. If *power*(t) = H and *status*(t) = `process` then the processor is in state *Processing*, if *power*(t) = H and *status*(t) = `not_process` then the processor is in state *Not processing and in high power state*, if *power*(t) = L and *status*(t) = `not_process` then the processor is in state *Not processing and in low power state*, and if *power*(t) = 0 and *status*(t) = `not_process` then the processor is in state *off*. Any other combination of values of variables *power*(t) and *status*(t) do not map to a processor's state.

To delimit the time horizon in which algorithms are evaluated we assume that any algorithm is notified by the adversary when the last job is released. This notification is received through a variable `OVER`(t) that takes value zero until the last release time of the instance and takes value one from there onwards. and to bound the total energy consumed we have created the state *off* which is just accessible by any algorithm once it has processed the entire instance.

4.2.1.4 LAZY

After introducing the notation to describe the dynamics of the processor, we are ready to introduce our procedure `LAZY`(t) used by LAZY to compute *status*(t) for every time $t \geq 0$. We define as S_{LAZY} the schedule return by algorithm LAZYfor a given instance I. The procedure `START`(t)

assumes that t is an integer time, however $status(t)$ can change at any time during the execution of the algorithm, when integrality is not assumed. Due to integrality assumption over release times and deadlines this does not present any inconvenient. To see why, let u be an arbitrary time in $(t, T_{\text{critic}}(t)]$. Advert that the output of $\texttt{START}(t)$ differs from the output of $\texttt{START}(u)$ only if an arrival occurs. Given that releases times and deadlines only occurs at integer times it follows that to know the value of $\texttt{START}(t)$ for an arbitrary time t is enough to compute $\texttt{START}(\lfloor t \rfloor)$.

For an arbitrary time $t \geq 0$, $\texttt{LAZY}(\text{t})$ receives as input $status(t^-)$ and all the available information regarding jobs that are released by time t (this include the value of the variable $\texttt{OVER}\,(t)$ at that time) , and returns $status(t)$. The variable $status(\text{t})$ changes from `process` to `not_process` when there is no workload to be processed during time step (t, t^+), and it changes from `not_process` to `process` at time $T_{\text{critic}}(t)$. To simplify the analysis, we assume that $status(0^-) = \texttt{process}$ for any algorithm.

4.2.2 Correctness of LAZY

In this section we show that in the schedule produced by algorithm LAZY for an arbitrary instance of problem energy($\rho_{\max}$,$s_{\max}$) all deadlines are met. We also show that starting to process after time $T_{\text{critic}}(t)$ the workload pending to be processed by algorithm LAZY at time t, no online algorithm can guarantee that all deadlines will be met.

We first introduce some definitions which are used to argue the correctness of algorithm LAZY. As defined in Section 4.2.1.4, S_{LAZY} is the schedule produces by algorithm LAZYfor an instance I. For an arbitrary schedule S, a *processing block*, is a maximal time interval where S is processing. An *idle period* is a maximal and finite time interval where S is not processing. Processing blocks are indexed in chronological order. The *cardinality* of a schedule S, $|S|$, is the number of processing blocks in schedule S. The j^{th} processing block of schedule S_{LAZY} is denoted as $B_j = \left(l_j^B, u_j^B\right]$ in which $j \geq 1$, we also define $l_0^B = 0$ and $u_0^B = 0$. To prove that in the schedule S_{LAZY} produce by $LAZY$ for an arbitrary instance of the problem energy($\rho_{\max}$,$s_{\max}$) all jobs meet their deadlines we use induction on the number of processing blocks in S_{LAZY}. If we consider an arbitrary processing block in S_{LAZY}, it follows clearly from definition of LAZY that no deadline that expires in $\left[u_{j-1}^B, l_j^B\right)$ is going to be missed. So to prove correctness of LAZY, it is enough to show that up to an arbitrary time in $\left[l_j^B, u_j^B\right)$ no deadline is missed.

Lemma 4.2.2. *Let I be an arbitrary instance of problem energy($\rho_{\max}$,$s_{\max}$), in the schedule resulting from applying algorithm LAZY on instance I all jobs meet their deadlines.*

Proof. We prove correctness of our algorithm LAZY by induction in the number of processing blocks in S_{LAZY}. Let $Y_j = \left(u^B_{j-1}, u^B_j\right]$ for all $1 \leq j \leq |S_{\text{LAZY}}|$.

- In the **base case** when at least one job has released time 0, it follows trivially that all jobs with release time in Y_1 meet their deadlines, because they are processed under earliest deadline first starting to process at time 0. If no job is release at time 0, assume for the sake of contradiction that a job with release time in Y_1 did not meet its deadline in S_{LAZY}. It implies that there exist a latest time $s < l^B_1$ such that starting to process at time s the deadline of all jobs pending to be processed at time s together with the deadlines of all jobs in an arbitrary augmented instance Js will be met. Given that $u^B_j > s$ then it follows that `LAZY`(s) returns `not_process`, which implies that `START`$(s) > s$. It implies that there is no augmented instance Js such that to meet the deadlines of jobs pending to be processed at time s and the deadlines of jobs in Js someone should start processing at time s, thus it follows a contradiction with definition of time s.

- In the **inductive step**, from induction hypothesis we may assume that all jobs with release time at most u^B_{j-1} have met their deadlines. We argue next that jobs with release time in Y_j meet their deadlines. Assume for the sake of contradiction that at least one job with release time in Y_j does not meet its deadline. Denote as s' earlier time in $\left(u^B_{j-1}, u^B_j\right]$ at which the total workload pending to be processed together with the workload that could be released by the adversary can not be processed without violating a deadline. Let $s = s' - 1$. From definition of s it follows clearly that starting to process at s all deadlines could be met, otherwise it will contradict the assumption that s' is the earliest time in Y_j at which if we have not started processing the pending workload at least one deadline will be violated. We may assume that s lies in time interval $\left(u^B_{j-1}, l^B_j\right)$, otherwise LAZY was already processing by time s contradicting the assumption that a deadline could be violated. From definition of routine `LAZY`(t) for some arbitrary time t together with definition of time s and the fact that at time s no job with release time larger than u^B_{j-1} is being processing it follows that `LAZY`(s) returns `process`, thus $s = l^B_j$, which contradicts the assumption that $s \in Y_j$. This conclude

the proof that in S_{LAZY} all jobs meet their deadlines.

□

4.3 Minimizing the number of idle periods

In this section we study the special case of a processor's power management mechanism in which the processor goes to a zero energy consumption power state when there is no workload pending to be processed. Also the processor consumes a non zero amount of energy, U, to power up and it consumes H power units per unit of time once is processing. We aim to minimize the total energy used to process an instance of problem energy(ρ_{max}, s_{max}). From the definition of this special case of processor's power management mechanism, it follows that the total energy required to process an arbitrary instance is equal to the energy required to process the workload plus the total energy consumed to restart processing. The total energy consumed to restart processing is equal to the total number of idle periods times the energy required to wake up the processor. As consequence, under this processor's power management mechanism the difference in the energy consumption between two algorithms, for the same instance, is completely determined by the number of idle periods in the resultant schedule and so it is independent of the length of the idle period.

Before presenting the competitive analysis of algorithm LAZY with performance measure the total number of idle periods, we will introduce some notation. Let S be a feasible schedule for an arbitrary instance of problem energy(ρ_{max},s_{max}). We denote as $N_{\text{idle}}(S)$ the number of idle periods of schedule S. We denote as $OPT_{\text{N}}(I)$ a feasible schedule of instance I with the smallest number of idle periods. We first argue that for the competitive analysis of algorithm LAZY we should only consider instances in which the optimal schedule has at least one idle period. This latter follows because for instances in which the optimal schedule has no idle period algorithm LAZY is optimal. To see why this is true, consider an arbitrary instance in which the optimal schedule has no idle period, then it follows that at any time instance from time 0 to the completion time of such schedule there is workload available to be processed. From the facts that at time 0 there is workload available to be processed (otherwise there is at least one idle period) and that from definition $u_0^B = 0$, it follows that algorithm LAZY will process at time 0 the workload that is

available and will stop processing just until there is no workload available to be processed. Thus it will processes at least until completion time of the optimal schedule, and so optimality of the schedule produces by LAZY follows.

Theorem 4.3.1. *Let I be an arbitrary instance of problem energy(1,$s_{\max}$) and let $S(I)$ be the schedule produces for such instance by an arbitrary online algorithm then:*

$$\max_I \frac{N_{\text{idle}}(S(I))}{N_{\text{idle}}(OPT_{\text{N}}(I))} \geq \begin{cases} \left\lfloor \frac{s_{\max}-5}{3} \right\rfloor + 2 & \text{for } s_{\max} \geq 5 \\ 2 & \text{for } s_{max} \in \{3,4\} \\ 1 & \text{for } s_{max} \in \{1,2\} \end{cases}$$

and LAZY is optimal for $s_{\max} = 1$.

Proof. We will first construct an instance I of a constrained adversary with parameters $\rho_{\max} = 1$, $s_{\max} \geq 5$, such that $N_{\text{idle}}(OPT_{\text{N}}(I)) = 1$ and such that any online algorithm will produce a schedule $S(I)$ with $N_{\text{idle}}(S(I)) \geq \left\lfloor \left| \frac{s_{\max}-5}{3} \right| \right\rfloor + 2$. Then we will prove that such instance is actually a valid instance, in other words is an instance that could have been released by the constrained adversary. Then we will conclude exposing an instance I of a constrained adversary with parameters $\rho_{\max} = 1$ and $s_{\max} \in \{2,3,4\}$ such that $N_{\text{idle}}(OPT_{\text{N}}(I)) = 1$ and such that any online algorithm will produce a schedule $S(I)$ with $N_{\text{idle}}(S(I)) \geq 2$. Finally we argue that for $s_{\max} = 1$, LAZY produces an optimal schedule.

Let `ALG'` be an arbitrary online algorithm with finite competitive ratio. We will first expose the instance I when $s_{\max} \geq 5$. Let us also define the following functions:

$$\begin{aligned} n_{\text{gaps}}(s_{\max}) &= \left\lfloor \frac{s_{\max}-5}{3} \right\rfloor + 2, \\ n_{\text{l}}(s_{\max}) &= 2 \cdot n_{\text{gaps}}(s_{\max}) - 3, \\ n_{\text{s}}(s_{\max}) &= n_{\text{gaps}}(s_{\max}), \\ n(s_{\max}) &= n_{\text{l}}(s_{\max}) + n_{\text{s}}(s_{\max}) + 1. \end{aligned}$$

At time 0 the adversary releases a job $J_0 = (1,0,1)$. Any algorithm should process it at its release time to satisfy the deadline. At times $i = 1, \ldots, n_{\text{l}}(s_{\max})$ the adversary released jobs $J_i = (1, i, s_{\max} + i)$ for $i = 1, \ldots, n_{\text{l}}(s_{\max})$. We claim that any online algorithm with a finite competitive ratio process all jobs $J_i, i = 1, \ldots n_{\text{l}}(s_{\max})$ as soon as they are released. Assume not. Then the adversary will not release any other job and will process all jobs J_i for $i = 1, \ldots, n_{\text{l}}(s_{\max})$

without idle period. Given that at least one of the jobs was not processed at its release time, then the schedule produces by algorithm `ALG'` has at least one idle period, thus `ALG''`s competitive ratio is not finite which is a contradiction.

At times $n_{\mathrm{l}}(s_{\max}) + 3 \cdot i - 1$ for $i = 1, \ldots n_{\mathrm{s}}(s_{\max})$ the constrained adversary released jobs:

$$J_{n_{\mathrm{l}}(d_{max})+i} = (1, n_{\mathrm{l}}(s_{\max}) + 3 \cdot i - 1, n_l(s_{\max}) + 3 \cdot i - 1), i \in 1, \ldots, n_s(s_{\max}).$$

From definition of such jobs, they are fully processed by any algorithm in time interval defined by its release time and deadline and given that these time intervals are not overlapping for any pair of these jobs, it follows that at least one idle period is produced. So any online algorithm will produce a schedule with at least $n_{\mathrm{s}}(s_{\max})$ idle periods for this instance meanwhile the optimal produces a schedule with one idle period. In figure 4.3.1 we show the previously defined instance when $s_{\max} = 9$. We can conclude that the schedule produces by any online algorithm with finite competitive ratio for such instance has at least $n_{\mathrm{s}}(s_{\max})$ idle periods meanwhile the optimal schedule has one idle period.

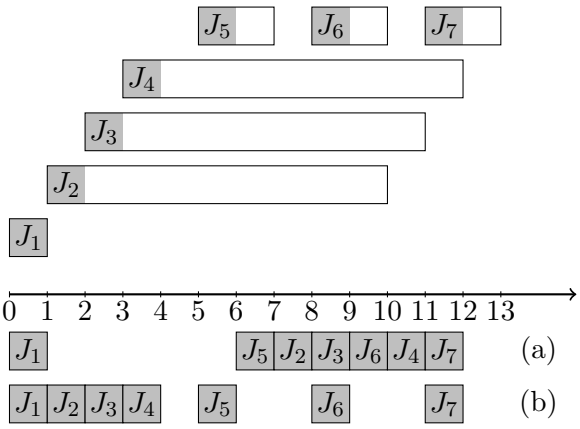


Figure 4.3.1: An instance of problem energy(1,9). (a) optimal schedule (b) schedule produces by an arbitrary online algorithm with finite competitive ratio.

For $s_{\max} \in \{3, 4\}$, the adversary releases a job $J_1 = (1, 0, s_{\max})$. Any online algorithm should process J_1 at its release time, otherwise the adversary will not send any other job and the algorithm has an unbounded competitive ratio which is a contradiction. At time $s_{\max} - 1$ the adversary releases job $J_2 = (1, s_{\max} - 1, s_{\max} + 1)$. Finally at time $s_{\max}$ the adversary releases job $J_3 = (1, s_{\max} + 1, s_{\max} + 2)$. Clearly this instance is an instance of a constrained adversary with

parameters $\rho_{max} = 1$ and $s_{\max}$ 3 or 4. In figure 4.3.2 we show the previously described instance when $s_{\max} = 4$. For this particular instance, J_2 should start to be processed at most at time 3 to guarantee that all deadlines will be met , and so at least one idle period occurs before the release time of J_3.

For $s_{\max} \in \{1, 2\}$, LAZY is an optimal algorithm, specifically, S_{LAZY} has the same number of

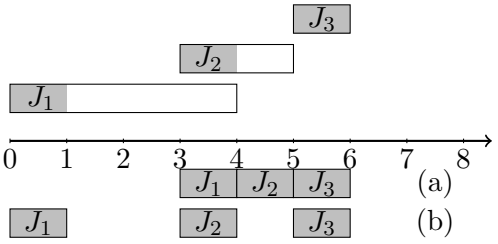


Figure 4.3.2: An instance of problem energy(1,4) (a) optimal schedule (b) schedule produces by an arbitrary online algorithm with finite competitive ratio.

idle periods than the optimal schedule. For $s_{\max} = 1$ optimality of S_{LAZY} follows trivially. We are just left to argue optimality for $s_{\max} = 2$. Let I be an arbitrary instance of problem energy(1,2) Assume that S_{LAZY} is not an optimal schedule. Let $[t, t+1)$ be the first time interval such that $OPT_{\text{N}}(I)$ or $LAZY$ are processing while the other one it is processing. Consider the case in which $[t, t+1)$ is an idle period in S_{LAZY}. From assumption that jobs are unitary, denote as J_1 the job that is processed in $OPT_{\text{N}}(I)$ in $[t, t+1)$. J_1 has release time $t-1$, which together with assumption that S_{LAZY} do not process in $[t, t+1)$ imply that J_1 is processed in $[t-1, t]$ in schedule S_{LAZY}. Given that the only job that could be pending to be processed at time t is J_1 in both schedules, it follows that $OPT_{\text{N}}(I)$ has processed more jobs up to time t than schedule S_{LAZY}, then it follows a contradiction with the definition of time t. The same argument holds for the case in which $[t, t+1)$ is an idle period in $OPT_{\text{N}}(I)$.

□

Theorem 4.3.2. *If I is an arbitrary instance of problem energy(1,$s_{\max}$), then:*

$$\max_I \frac{N_{\text{idle}}(S_{\text{LAZY}}))}{N_{\text{idle}}(OPT_{\text{N}}(I))} \leq \frac{s_{\max}}{2} + 2.$$

Proof. Let I be an arbitrary instance of problem energy(1,s_{max}). From definitions, S_{LAZY}, $OPT_{\text{N}}(I)$ are instance's I schedule produces by $LAZY$ and the schedule with minimum number of idle pe-

riods, respectively. We denote as $\{C_l\}$ the set of processing blocks of schedule $OPT_{\mathrm{N}}(I)$. Let us define the following sets for each processing block C_l :

$$\begin{aligned} N_1^l &= \left\{B_k \in S_{\mathrm{LAZY}} : u_{l-1}^C \leq u_k^B < l_l^C\right\} \quad \forall l \geq 1, \ldots, N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I)), \\ N_2^l &= \left\{B_k \in S_{\mathrm{LAZY}} : l_l^C \leq u_k^B < u_l^C\right\} \quad \forall l \geq 1, \ldots, N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I)). \end{aligned}$$

From definition of sets N_1^l and N_2^l it follows that:

$$\begin{aligned} N_{\mathrm{idle}}(S_{\mathrm{LAZY}}) &\leq \left[\left(\sum_{l=1}^{N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I))} |N_1^l| + |N_2^l|\right) + 1\right] - 1 \qquad (4.3.1) \\ &= \sum_{l=1}^{N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I))} |N_1^l| + |N_2^l| \\ &\leq N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I)) \cdot \left[\max_{l=1,\ldots,|OPT_{\mathrm{N}}(I)|} |N_1^l| + |N_2^l|\right]. \end{aligned}$$

The $+1$ in the first inequality counts a processing block of S_{LAZY} that may terminates after the last processing block of $OPT_{\mathrm{N}}(I)$ which is not in $N_2^{N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I))}$ from definition. The -1 comes from the fact that the number of gaps is the number of processing blocks minus 1 (by default we have assume that there is a processing block that started at time 0^- not preceded by an idle period). We will show:

$$\max_{l=1,\ldots,|OPT_{\mathrm{N}}(I)|} |N_1^l| + |N_2^l| \leq \frac{s_{\max}}{2} + 2, \qquad (4.3.2)$$

which imply that $\frac{N_{\mathrm{idle}}(S_{\mathrm{LAZY}})}{N_{\mathrm{idle}}(OPT_{\mathrm{N}}(I))} \leq \frac{s_{\max}}{2} + 2$, which is the desired result. We will denote from now onwards as l^* the index in $1, \ldots, |OPT_{\mathrm{N}}(I)|$ that maximizes the LHS of inequality (4.3.2). To prove inequality (4.3.2) we first we argue that the length of any idle period that precedes a processing block $B_j \in N_2^{l^*}$ but the last one has length at least 2. Next we argue that the length of any processing block in $N_1^{l^*}$ but the first one is at least 2.

Claim 1. *For every $B_j \in N_2^{l^*}$ but the last one it holds that $l_j^B - u_{j-1}^B \geq 2$.*

Proof. Let B_j be a processing block in $N_2^{l^*}$ but the last one. Assume for the sake of contradiction that $l_j^B - u_{j-1}^B = 1$. We may assume that at time l_j^B schedule $OPT_{\mathrm{N}}(I)$ has pending workload with release time at least $u_{j-1}^B - 1$, otherwise it contradicts definition of $LAZY$ (if release time is u_{j-1}^B) or it implies that B_j is the last processing block in $N_2^{l^*}$ (because both algorithms at time l_j^B

will have available the same amount of workload to process) which is a contradiction. Thus the density profile from time l_j^B and onwards takes at least one non one value affected by a job with released time at most u_{j-1}^B and already processed in S_{LAZY}, and given that S_{LAZY} has no pending workload immediately before time $u_{j-1}^B + 1$ (which from assumption is equal to l_j^B) , algorithm LAZY will start processing at least at time $l_j^B + 1$ and guarantee that all deadlines will be met, which contradicts assumption that LAZY starts processing at time l_j^B. □

Claim 2. *For every processing block $B_j \in N_1^{l^*}$ but the first one, $l_j^B - u_{j-1}^B \geq 2$.*

Proof. Assume not. There exist a processing block $B_j \in N_1^{l^*}$ such that $u_j^B - l_{j-1}^B = 1$. This implies that a job J_1 is released at time $u_{j-1}^B + 1$ and it should be processed by $LAZY$ at its release time. There was no workload pending to be processed in $OPTI$ just before time l_{j-1}^B, which implies that B_j is the first processing block in $N_1^{l^*}$ which is a contradiction. So we may assume that there is at least one job pending to be processed in $OPT_{\text{N}}(I)$ at time l_j^B and already processed in S_{LAZY}. The density profile from time l_j^B and onward takes at least one non one value, which implies that LAZY could start processing at time $l_j^B + 1$ and still meet all deadlines. This contradicts definition of l_j^B. □

We now complete the proof of Theorem 4.3.1. Let us denote as $\phi(C_{l^*})$ the amount of workload that is pending to be processed in $OPT_{\text{N}}(I)$ and not in S_{LAZY} at time $l_{l^*}^C$. From the first release time in time interval $(u_{l^*-1}^C, u_{l^*}^C]$ to time $l_{l^*}^C$ there are at most $s_{\max}$ time units and so S_{LAZY} process during at most $s_{\max}$ time units minus the time that it was idle. This implies that at time $l_{l^*}^C$ there are at most $s_{\max} - 2\left(|N_1^{l^*}| - 1\right)$ workload units pending to be processed in $OPT_{\text{N}}(I)$ and already processed in S_{LAZY}. In other words $\phi(C_{l^*}) \leq s_{\max} - 2\left(|N_1^{l^*}| - 1\right)$.

Given that $OPT_{\text{N}}(I)$ is processing during all time intervals $[u_{j-1}^B, l_j^B]$, $\forall B_j \in N_2^{l^*}$ but the first one. Then total workload processed by $OPT_{\text{N}}(I)$ in intervals $[u_{j-1}^B, l_j^B]$, for all $B_j \in N_2^{l^*}$ but the first one, is bounded above by $\phi(C_{l^*})$. From Claim 1 it follows that $l_j^B - u_{j-1}^B \geq 2$ for all $B_j \in N_2^{l^*}$ but the first one, this fact together with the assumption that in $OPT_{\text{N}}(I)$ workload is in process during all these time intervals and so there are at most $\phi(C_{l^*})/2 + 1$ idle periods in $N_2^{l^*}$. putting this information together it follows that we can upper bound the sum of the number of idle periods

in $N_1^{l^*}$ and $N_2^{l^*}$ by:

$$\begin{aligned} |N_1^{l^*}| + |N_2^{l^*}| &\leq |N_1^{l^*}| + \frac{\phi(C_{l^*})}{2} + 1 \\ &\leq |N_1^{l^*}| + \frac{s_{\max} - 2 \cdot \left(|N_1^{l^*}| - 1\right)}{2} + 1 \\ &= \frac{s_{\max}}{2} + 2 \end{aligned} \tag{4.3.3}$$

Then replacing inequality (4.3.3) in inequality (4.3.1) it follows:

$$\begin{aligned} N_{\text{idle}}(S_{\text{LAZY}}) &\leq \sum_{h=1}^{|OPT_{\text{N}}(I)|-1} |N_1^h| + |N_2^h| \\ &= \left(\frac{s_{\max}}{2} + 2\right) \cdot N_{\text{idle}}(OPT_{\text{N}}(I)) \end{aligned}$$

which is the desired result. □

4.4 Energy minimization

In this section we present our energy minimization algorithm and its competitive analysis. In Section 4.4.1 we present our energy minimization algorithm *DEFER*. Next in Section 4.4.2 we present its competitive analysis as well as a lower bound for an arbitrary online algorithm that guarantee that all jobs meet their deadlines. We provide asymptotic lower and upper bounds as a function of the maximum span parameter $s_{\max}$. The bounds that we provide are the result of a worst case analysis and thus they are independent of the processor's energy parameters, H, L and U.

4.4.1 Algorithm DEFER

4.4.1.1 Preliminaries

An online algorithm for problem energy($\rho_{\max}$,$s_{\max}$) takes two type of decisions: 1) when and what to process? 2) When not processing, at which power level the processor should be operating? In Section 4.2 we presented our online scheduling algorithm LAZY, which we use as a subroutine of algorithm DEFER to decide when and what to process. Algorithm DEFER basically schedules jobs using the scheduling algorithm LAZY and manages the energy during the idle periods using an

optimal 2 competitive deterministic online algorithm, see [9] for a detailed description of an optimal 2 competitive deterministic online algorithm. Recall that the we are assessing the performance of an algorithm from a worst case perspective.

We first introduce some new definitions that will be useful to describe the results in this section. For an arbitrary schedule S and a time t. We define $Y(t, S)$ as the machine's energy consumption rate at time t for a given schedule S. We also define $N_{\text{energy}}(S)$ as the total energy consumed by a machine with energy parameters H, L and U that processes schedule S and manages idle periods using an optimal 2 competitive deterministic online algorithm. For an arbitrary instance I we define $OPT_{\text{e}}(I)$ as a schedule that minimize the total energy consumed to process instance I. Finally for an arbitrary schedule S we define a *short* idle period as an idle period of S with length l such that $H \cdot l \leq L \cdot l + U$. A long idle period is an idle period of S with length l such that $H \cdot l > L \cdot l + U$.

4.4.1.2 The algorithm

In this section we present our online energy minimization algorithm DEFER. As defined in Section 4.1, the processor is a single machine with two power states. H and L are the energy consumed per unit of time processing in high and low power state respectively. We also defined U as the energy consumed during a transition from low power state to high power state and we assume that transitions occur in a negligible fixed about of time. The processor processes on high power state: it consumes H energy units per unit of workload that it processes; and when it is not processing it consumes energy at high or at low power state. Next we described the online algorithm DEFER. For an arbitrary instance I,

- Schedule jobs following the scheduling decision taken by algorithm LAZY.
- Let S_{LAZY} be the schedule produces by $LAZY$. For an arbitrary time t, if S_{LAZY} is processing a job let $power(t) = H$. Otherwise, if δ is the elapsed time since the start of current idle period, B_i is the processing block of S_{LAZY} that preceeds current idle period and $H \cdot \delta \geq L \cdot \delta + U$ let $power(u_i^B + \delta) = L$, otherwise $power(u_i^B + \delta) = H$.

Notice that DEFER takes scheduling decisions independently of the processor's energy parameters. We argue next that once a machine is idle, the decision of when to start processing can be done

independently of the energy parameters of the processor (H, L, U). Notice that the value of the processor's energy parameters does not affect the load capacity of the adversary, more specifically the jobs that could be released by the adversary are independent of the energy parameters of the processor. It implies that the ability of an algorithm to forecast at current time a lower bound on the length of the very next idle period (or the length of current idle period) is independent of the processor's parameters. Recall that we are looking at this problem from a worst case perspective and that we would like to provide an lower(upper) bound on the ratio between the total energy consumed by an arbitrary online algorithm (our online algorithm) independent of the processor's energy parameters and function of the maximum job span parameter $s_{\max}$.

Consider the case in which the scheduler is not processing and there is pending workload. Using the same optimization problem used in by algorithm LAZY , Problem 1, someone could find an upper bound on the maximum length of the current idle period (time that has elapsed in current idle period plus maximum time left to start processing to guarantee that all deadlines will be met). For some processor's energy parameters this upper bound corresponds to a short idle period and for some others it corresponds to a long idle period. Consider the case in which an algorithm starts processing before it is mandatory in order to guarantee that all deadlines will be met, equivalently, this algorithm makes current idle period shorter than the upper bound on the length of such idle period. If according to the processor's energy parameters the maximum length of this idle period corresponds to the length of a long idle period, then starting to process earlier will produce in the worst case a short idle period and a short/long one with total length at least the length of an long idle period, which is locally less energy efficient than a long idle period. If according to the processor's energy parameters the maximum length of this idle period corresponds to the length of a short idle period, then starting to process earlier will produce in the worst case a short idle period and another short/long idle period with total length at least the upper bound on the length of current idle period. In both cases, someone is locally better off by starting to process as late as possible.

Intuitively, pending workload may allow a scheduler to process a set of jobs that will be released in the future uninterruptedly that otherwise will be forced to process with an idle period in between. More idle periods are associated with more waste of energy. Thus we could see the ability to hold workload to process it in the future as a resource that erodes the power of the adversary to force

an online algorithm to create more idle periods in order to meet all deadlines.

4.4.2 Competitive Analysis

In this Section we present the competitive analysis of algorithm DEFER. In Theorem 4.4.1 we present an asymptotic lower bound in the competitive analysis of an arbitrary online algorithm when $s_{\max}$ goes large, and in Proposition 4.4.2 we present an asymptotic upper bound in the competitive analysis of our online algorithm DEFER when $s_{\max}$ goes large.

Theorem 4.4.1. *Let* ALG *be an arbitrary online algorithm that manages power consumption during idle periods using an optimal competitive algorithm. If I is an arbitrary realization of problem energy($\rho_{\max}$,$s_{\max}$) and S is the schedule produces by algorithm* ALG *for instance I, then:*

$$\max_{I} \frac{N_{\text{energy}}(S)}{N_{\text{energy}}(OPT_{\text{e}}(I))} \quad = \quad \Omega\left(\ln s_{\max}\right)$$

when $\rho_{\max} = 1$.

Proof. We may assume without loss of generality that preemption is not allowed because allowing preemption could only increase the power of the adversary, thus allowing the lower bound on the competitive ratio of an arbitrary online algorithm to increase. Notice that we will show an asymptotic lower bound on $s_{\max}$ then we may assume that $s_{\max}$ is a large number.

Let us consider the following instance of problem energy(1,$s_{\max}$). Given that we are performing an asymptotic analysis, when $s_{\max}$ goes large, we may assume that $s_{\max} = 2 \cdot 3^M$ for some integer M, ($s_{\max} \geq 5$.

$$\begin{aligned} j_1 &= (2, 2, 2 + s_{max}) \\ j_i &= \left(2, s_{max} + 2 \cdot (i-1) - \tfrac{s_{max}}{2 \cdot 3^{i-2}} + 2, s_{max} + 2 \cdot i\right) \quad \text{for } 2 \leq i \leq \left\lfloor \tfrac{\ln \frac{s_{max}}{4}}{\ln 3} + 2 \right\rfloor. \end{aligned}$$

Let i be an arbitrary index in $1, \ldots, \left\lfloor \frac{\ln \frac{s_{\max}}{4}}{\ln 3} + 2 \right\rfloor$. To prove this lower bound we will first proceed to gain information about the schedule produces by an arbitrary online algorithm. Let $m_i = s_{\max} + 2 \cdot i - \frac{s_{\max}}{2 \cdot 3^{i-1}}$ We will first argue that any online algorithm should process at least one workload unit of job i in time interval $[r_i, m_i + 1]$, to guarantee that all deadlines will be met. Assume not. At time $m_i + 1$ a job with workload $\frac{s_{\max}}{2 \cdot 3^{i-1}} - 1$, and deadline $s_{\max} + 2 \cdot i$ could be released, and given that for $i \geq 2$:

$$\begin{aligned} m_i + 1 + \frac{s_{\max}}{2 \cdot 3^{i-1}} - 1 + 2 &= s_{\max} + 2 \cdot i - \frac{s_{\max}}{2 \cdot 3^{i-1}} + \frac{s_{\max}}{2 \cdot 3^{i-1}} - 1 + 2 \\ &> s_{\max} + 2 \cdot i \end{aligned} \tag{4.4.1}$$

Then the algorithm will not be able to meet all deadlines. Similarly for $i = 1$ inequality (4.4.1) follows.

Let us denote as u_i the number of idle units of time in interval $[r_i - 1, m_i + 1]$ for an arbitrary job index i in the instance. Notice that $m_i + 1 = r_{i+1} - 1$. Let us denote as H, 0 and U the energy consumption rate at high power state, at low power state and the total energy consumed during a transition from low power state to high power state respectively. Let us define $t = \frac{U}{H}$. From definition of mechanism DEFER, if the idle period has length at least t units, then the processor consumes energy at the rate of H during the first t time units of the idle period and then it powers down. It is because t is the solution to the equation $H \cdot t = 0 \cdot t + U$ in which the energy consumed at high power rate is equal to the energy necessary to power up. Notice that under this particular set of processor's energy parameters, no energy is consumed while the processor is in low power state. We may assume for this analysis that $H = 1$ and thus $t = U$. The reason why is that we are interest in the ratio $\frac{U}{H}$ and not on their individual values. As a consequence of the fact that at least one workload unit is processed during time interval $[r_i, m_i + 1]$ and preemption is not allowed it follows that:

$$u_i = \begin{cases} \frac{s_{\max}}{2} & \text{If } i = 1 \\ \frac{s_{\max}}{3^{i-1}} & \text{If } i \geq 2. \end{cases}$$

Let us define S as the schedule such that:

- *If $u_i > t$, i is odd and $u_{i+1} \geq t$:* Start to process job i at time r_i.
- *If $u_i > t$, i is odd and $u_{i+1} < t$*: If $2\,u_i \geq t$ Start to process job i at time r_i, otherwise at time $r_i + 2\,u_{i+1}$.
- *If $u_i > t$ and i is even:* Start to process job i at time m_i.
- *If $u_i \leq t$:* Start to process job i at time m_i.

We claim that no online algorithm that manages power consumption during an idle period using an optimal competitive algorithm can consume less energy than S. Assume not. Let S' be the schedule obtained with an online algorithm and assume that S' consumes less energy than S. Given that both algorithms spend the same energy processing workload then S' spend less energy than S during idle periods. Define interval T_i as $T_i = [r_i - 1, m_i + 1]$ for each job index in the instance under consideration. We will argue that the energy spent by S is at most energy spend by S'. Let us consider the energy consumption of any schedule S' respect to schedule S at each interval T_i and its impact in the overall energy consumption.

- *If $u_i > t$, i is odd and $u_{i+1} \geq t$:* Given that $u_i \geq t$, any algorithm consume at least t units of energy during idle periods in time interval $[r_i, m_i + 1]$. From definition of S at time $r_i + 2$ an idle period of length at least t starts and it doesn't terminate in time interval T_i.

- *If $u_i > t$, i is odd and $u_{i+1} < t$:* If $2 \cdot u_{i+1} < t$, given that $\frac{u_i + u_{i+1}}{2} = 2 \cdot u_{i+1} < t$ it follows that $u_i + u_{i+1} < 2\,t$, the total energy consumed in $T_i \cup T_{i+1}$ when job i is processed at time $r_i + 2\,u_{i+1}$ is equal to $u_i + u_{i+1} < 2\,t$ which is the smaller amount of energy that any online algorithm could have consumed in $T_i \cup T_{i+1}$. In case, $2\,u_{i+1} > 2\,t$, it follows that $u_i + u_{i+1} > t$, thus processing job i at time r_i guarantee that the energy consumed in $T_i \cup T_{i+1}$ is $2\,t$ which is a lower bound on the energy consumed by any online algorithm that use an optimal competitive algorithm to manage the idle periods in a total idle length which is strictly greater than $2\,t$.

- *i is even and $u_i \geq t$:* Given that $u_i \geq t$, any algorithm consume at least t units of energy during idle periods in time interval $[r_i, m_i + 1]$. From definition of S and definition of u_i it follows that at time m_i an idle period of length at least $2 \cdot t$ terminates. Given that $u_{i-1} \geq t$ and job j_{i-1} is started to be processed at time r_i it follows that during idle period in time interval $[r_i, m_i + 1]$ the energy consumed by S is the energy consume to transit from low power state to high power state, which is t.

- *If $u_i < t$:* Then it follows that any algorithm consumes at least u_i units of energy during idle periods in time interval $[r_i, m_i + 1]$. Given that u_i is the minimum energy that any online algorithm could consume during time interval $[r_i, m_i + 1]$ and S consumes u_i energy units during such time interval. Then it follows that S' cannot do better.

Notice that given that at each T_i at least one job is processed, it follows that optimality in all pair of intervals $T_i \cup T_{i+1}$ implies global optimality o energy consumed by schedule S.

Now we are ready to establish a lower bound of the energy consumed in schedule S as a function of t and having $s_{\max}$ as a parameters. From definition of u_i, it is a decreasing function in i. Let us define $\bar{i}$ as the largest i such that $u_i \geq t$. From definition of u_i it follows that:

$$\bar{i} \leq \frac{\ln \frac{s_{\max}}{t}}{\ln 3} + 1$$

Let us define l as $\left\lfloor \frac{\ln \frac{s_{\max}}{t}}{\ln 3} + 1 \right\rfloor$. From its definition l is the number of job indexes such that $u_i \geq t$. Let us define n as $\left\lfloor \frac{\ln \frac{s_{\max}}{4}}{\ln 3} + 2 \right\rfloor$. From its definition n is the total number of jobs in instance under consideration. Using definition of l and n, we establish a lower bound on the energy consumed by S, $N_{\text{energy}}(S)$:

$$\begin{aligned}
N_{\text{energy}}(S) &\geq (t \cdot l + 1) + \sum_{j=l+1}^{n} u_j + 2 \cdot n \\
&= t \cdot l + \sum_{j=l+1}^{n} \frac{s_{\max}}{3^{i-1}} + (2 \cdot n + 1) \\
&= t \cdot l + \frac{s_{\max}}{2} \cdot \left[\left(\frac{1}{3}\right)^{l} - \left(\frac{1}{3}\right)^{n} \right] + 2 \cdot n + 1 \\
&\geq \frac{t}{\ln 3} \cdot \ln \frac{s_{\max}}{t} + \frac{1}{6} \cdot t + \left[-\frac{2}{3} + 2 \cdot n + 1 \right] \\
&= \frac{t}{\ln 3} \cdot \ln \frac{s_{\max}}{t} + \frac{t}{6} + \left[2 \cdot n + \frac{1}{3} \right] \\
&\geq \frac{t}{\ln 3} \cdot \ln \frac{s_{\max}}{t} + \frac{t}{6} + \left[\frac{2}{\ln 3} \cdot \ln \frac{s_{\max}}{4} + \frac{4}{3} \right],
\end{aligned}$$

in which the second inequality follows from the fact that $\frac{s_{\max}}{2} \cdot \left(\frac{1}{3}\right)^{l} \geq \frac{t}{6}$, because from definition of l, $u_l = \frac{s_{\max}}{3^{l-1}} \geq t$, which through simple transformations lead us to the desired result. Similarly, $-\frac{s_{\max}}{2} \cdot \left(\frac{1}{3}\right)^{n} \geq -\frac{2}{3}$ because from definition of n, $d_{n+1} - r_{n+1} = \frac{s_{\max}}{2 \cdot 3^{n-1}} < 2$, which through simple manipulations give us the desired result. On the other hand, from definition of the instance $N_{\text{energy}}(OPT_{\text{e}}(S)) = t + 2 \cdot n + 1$.

In order to find a lower bound on the competitive ratio of an arbitrary online algorithm, we would like to maximize the RHS of inequality (4.4.2).

$$\frac{N_{\text{energy}}(S)}{OPT_{\text{e}}(S)} \geq \frac{\frac{t}{\ln 3}\cdot \ln\frac{s_{\max}}{t} + \frac{t}{6} + \left[\frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + \frac{4}{3}\right]}{t + \frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3}. \tag{4.4.2}$$

Claim 1.

$$\max_t \frac{N_{\text{energy}}(S)}{OPT_{\text{e}}(S)} \geq \frac{a\cdot \ln^{b+1} s_{\max} - a\cdot b\cdot \ln^b s_{\max}\cdot \ln\ln s_{\max}}{a\cdot \ln^b s_{\max} + \frac{2}{\ln 3}\cdot \ln s_{\max} - \frac{2\cdot \ln 4}{\ln 3} + 3} - \frac{\left(a\cdot \ln a - \frac{a}{6}\right)\cdot \ln^b s_{\max} + \frac{2}{\ln 3}\cdot \ln s_{\max} + \left(\frac{4}{3} - \frac{2\cdot \ln 4}{\ln 3}\right)}{a\cdot \ln^b s_{\max} + \frac{2}{\ln 3}\cdot \ln s_{\max} - \frac{2\cdot \ln 4}{\ln 3} + 3},$$

for some $0 < a < \frac{2}{\ln 3}$ *and* $0 < b < 2$.

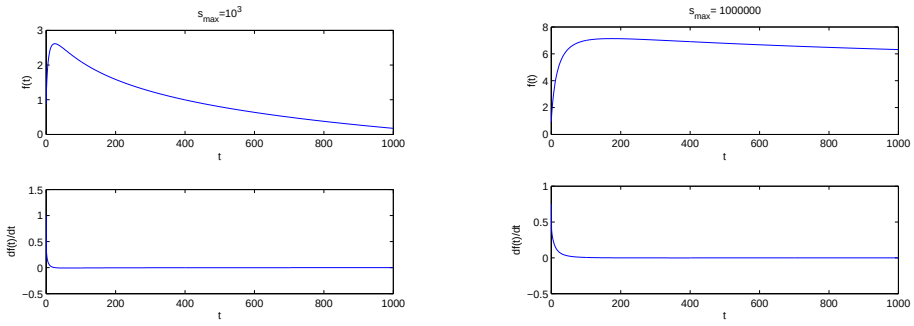
Proof. Let us define $f(t)$ as:

$$f(t) = \frac{\frac{t}{\ln 3}\cdot \ln\frac{s_{\max}}{t} + \frac{t}{6} + \left[\frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + \frac{4}{3}\right]}{t + \frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3}.$$

We will first argue that $f(t)$ has a global maximum. Let us consider $f'(t)$:

$$f'(t) = \frac{\left(\frac{1}{6} - \frac{1}{\ln 3}\right)\cdot\left(\frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3\right) - \frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} - \frac{4}{3}}{\left(t + \frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3\right)^2} + \frac{\frac{1}{\ln 3}\cdot\left(\frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3\right)\cdot \ln\frac{s_{\max}}{t} - \frac{t}{\ln 3}}{\left(t + \frac{2}{\ln 3}\cdot \ln\frac{s_{\max}}{4} + 3\right)^2}$$

From definition of $f'(t)$, it follows clearly that $f'(t)$ is a decreasing function for $0 < t < s_{\max}$. In Figure 4.4.1 we present $f(t)$ and $f'(t)$ for $s_{\max} = 10^3$. Now we will argue that for $t_1 = \epsilon$ in which



(a) $s_{\max} = 10^3$ (b) $s_{\max} = 10^6$

Figure 4.4.1: $f(t)$ and $f'(t)$ for a) $s_{\max} = 10^3$ b) $s_{\max} = 10^6$

$\epsilon > 0$ is a very small number, $f'(t_1) > 0$. And that for $t_2 = s_{\max}$, $f'(t_2) < 0$. This fact implies that $f(t)$ has a global maximum in domain $0 < t < s_{\max}$. Given that denominator of $f'(t)$ is positive in all the domain of $f(t)$ we just need to argue that the numerator of $f(t)$ is positive for $t = t_1$ and is negative for $t = t_2$. Let us denote the numerator of $f'(t)$ as $n_{f'}(t)$, which can be written as:

$$\begin{aligned} n_{f'}(t) &= \ln\frac{s_{\max}}{4}\cdot\left[\frac{2}{\ln 3}\cdot\frac{1}{6}-\frac{2}{\ln^2 3}-\frac{2}{\ln 3}+\frac{2}{\ln^2 3}\cdot\ln\frac{s_{\max}}{t}\right] \\ &\quad +\left[-\frac{3}{\ln 3}-\frac{4}{3}+\frac{3}{\ln 3}\cdot\ln\frac{s_{\max}}{t}-\frac{t}{\ln 3}\right]. \end{aligned}$$

Given that $\ln\frac{s_{\max}}{\epsilon}$ goes to infinity as ϵ goes to 0. It follows that $n_{f'}(\epsilon) > 0$. On the other hand, at $t = s_{\max}$ it follows that:

$$\begin{aligned} n_{f'}(s_{\max}) &= \ln\frac{s_{\max}}{4}\cdot\left(-\frac{5}{3\cdot\ln 3}-\frac{2}{\ln^2 3}\right)+\left(-\frac{3}{\ln 3}-\frac{4}{3}-\frac{s_{\max}}{\ln 3}\right) \\ &\leq 0. \end{aligned}$$

Then we can conclude that $f(t)$ has a unique maximizer in $0 < t < s_{\max}$. Let us denote as $t^*(s_{\max})$ the value of t that maximizes $f(t)$ for a given $s_{\max}$. Let us define $l(s_{\max}) = a\cdot\ln^b(s_{\max})$ for some constants $0 < a < \frac{2}{\ln 3}$, $0 < b \leq 2$. We will argue that $t^* = \Omega(l(s_{\max}))$. To argue this, is enough to show that when $s_{\max}$ is very large, $n_{f'}(t)$ for $t = a\cdot\log^b(s_{\max})$, is nonnegative. Although we don't know how to compute a closed form solution of $t^*(s_{\max})$ we can compute numerically its value for a given value of $s_{\max}$. In Figure 4.4.2 we present $t^*(s_{\max})$ as a function of $s_{\max}$. Also the fitted curve resulted from fitting $t^*(s_{\max})$ by $a\ln{s_{\max}}^b$ for $b = 1.9$ and $b = 2.1$. After some

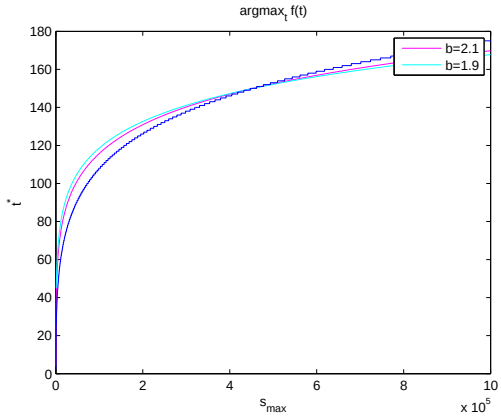


Figure 4.4.2: $t^*(s_{\max})$ computed numerically

manipulation it follows that:

$$\begin{aligned} n_{f'}\left(a\cdot\ln^b(s_{\max})\right) &= \left[\left(\frac{1}{\ln 3}-\frac{1}{6}\right)\cdot\frac{2\cdot\ln 4}{\ln 3}-3\cdot\left(\frac{1}{\ln 3}-\frac{1}{6}\right)\right. \\ &\quad \left.+\frac{2\cdot\ln 4}{\ln 3}-\frac{4}{3}-\frac{3\cdot\ln a}{\ln 3}+\frac{2\cdot\ln 4\cdot\ln a}{\ln 3}\right] \\ &\quad +\ln s_{\max}\cdot\left[-\left(\frac{1}{\ln 3}-\frac{1}{6}\right)\cdot\frac{2}{\ln 3}-\frac{2}{\ln 3}\right. \\ &\quad \left.-\frac{2\cdot\ln 4}{\ln 3}+\frac{3}{\ln 3}-\frac{2\cdot\ln a}{\ln^2 3}+\left(-\frac{3}{\ln 3}+\frac{2\cdot\ln 4}{\ln^2 3}\right)\cdot b\right] \\ &\quad +\ln^2 s_{\max}\cdot\frac{2}{\ln^2 3} \\ &\quad -\ln s_{\max}\cdot\ln\ln s_{\max}\cdot\frac{2\cdot b}{\ln^2 3} \\ &\quad -\frac{a}{\ln 3}\cdot\ln^b s_{\max}. \end{aligned}$$

Given that $b<2$ it follows that $n_{f'}\left(a\cdot\log^b s_{\max}\right)$ is nonnegative as $s_{\max}$ goes large.
Now let us define $u(s_{\max})=c\cdot\ln^d(s_{\max})$, $c>0$, $d>2$. In the same order of ideas used to argue that $t^*=\Omega(l(s_{\max}))$ it follows that for $d>2$, $t^*=O(u(s_{\max}))$.

$$\begin{aligned} f(t^*) &\geq \frac{a\cdot\ln^{b+1}s_{\max}-a\cdot b\cdot\ln^b s_{\max}\cdot\ln\ln s_{\max}}{a\cdot\ln^b s_{\max}+\frac{2}{\ln 3}\cdot\ln s_{\max}-\frac{2\cdot\ln 4}{\ln 3}+3} \\ &\quad -\frac{\left(a\cdot\ln a-\frac{a}{6}\right)\cdot\ln^b s_{\max}+\frac{2}{\ln 3}\cdot\ln s_{\max}+\left(\frac{4}{3}-\frac{2\cdot\ln 4}{\ln 3}\right)}{a\cdot\ln^b s_{\max}+\frac{2}{\ln 3}\cdot\ln s_{\max}-\frac{2\cdot\ln 4}{\ln 3}+3} \end{aligned}$$

□

From Claim 4.4.2 it follows that when $s_{\max}$ goes large $\frac{N_{\text{energy}}(S)}{OPT_{\text{e}}(S)}=\Omega(\ln s_{\max})$, which is the desired result. □

Next we present in Theorem 4.4.2 an asymptotic upper bound for the schedule produces by algorithm DEFER. As reminder, algorithm DEFER schedules jobs using algorithm LAZY and manages the energy during idle periods using a 2 deterministic optimal online algorithm. In Proposition 4.4.1 we prove that for an arbitrary instance I we could always find an instance $\hat{I}$ in which the optimal schedule has at most 3 idle periods such that the competitive ratio for instance I is at most 3 times the competitive ratio of instance $\hat{I}$. We use the results in Proposition 4.4.1 and in its Claim 2 in the proof of Theorem 4.4.2.

Proposition 4.4.1. *For every arbitrary instance I of the constrained adversary with parameters $s_{\max}$ and $\rho_{\max} = 1$ which is processed by a processor with energy parameters (H, U), such that $U > H$, there exist an instance $\hat{I}$ for which the optimal schedule has at most 3 idle periods, such that:*

$$\max_I \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))} \leq 3 \cdot \frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}.$$

Additionally, if $OPT_{\text{e}}(\hat{I})$ has 3 idle periods, then there is no processing block of the schedule $\hat{S_{\text{LAZY}}}$ that has time overlap neither with the first, neither with the last idle period of schedule $OPT_{\text{e}}(\hat{I})$. If $OPT_{\text{e}}(I)$ has 2 idle periods, then there is no processing block of the schedule $\hat{S_{\text{LAZY}}}$ that has time overlap with the first idle period of schedule $OPT_{\text{e}}(\hat{I})$.

Proof. Let I^* be an instance in $\arg\max_I \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))}$ with the smallest total workload. Let $OPT_{\text{e}}(I^*)$ be an optimal schedule of instance I such that workload is processed as early as possible. In other words, in which no workload unit could have been processed later without loosing optimality.

Claim 2. *The schedule $OPT_{\text{e}}(I^*)$ has no short idle period.*

Proof. Assume not. From assumption there exist at least one "short" idle period in $OPT_{\text{e}}(I^*)$. Let us denote as I_s the first "short" idle period in $OPT_{\text{e}}(I^*)$ and let us denote as C_s the very next processing block. We argue first that the workload processed in C_s have release time at least l_s^C. Assume not. Let us consider an arbitrary workload unit processed in processing block C_s with release time strictly prior to l_s^C. Let us define S as the schedule that result from schedule $OPT_{\text{e}}(I^*)$ but processing such workload unit at time $l_s^C - 1$. The energy consumed by schedule S is at most the energy consumed by schedule $OPT_{\text{e}}(I^*)$, $N_{\text{energy}}(OPT_{\text{e}}(I^*))$, then we can conclude that schedule S is also an optimal schedule of instance I^* and one workload unit is processed before it is processed in $OPT_{\text{e}}(I^*)$, which contradicts assumption about $OPT_{\text{e}}(I^*)$. Thus, we conclude that the total workload processed in C_s have release time at least l_s^C.

Now we will argue that workload pending to be processed in schedule $OPT_{\text{e}}(I^*)$ by time u_s^C has release time at least l_s^C. Assume not. Let us consider an arbitrary workload unit that haven't been processed in $OPT_{\text{e}}(I^*)$ by time u_s^C. By assumption there is at least one workload unit with release time strictly smaller than l_s^C. Let us consider a schedule equal to $OPT_{\text{e}}(I^*)$ but such workload unit

is processed at time $l_s^C - 1$, the energy consumed by the resulting schedule is a lower bound on $N_{\text{energy}}(OPT_{\text{e}}(I^*))$ and given that $N_{\text{energy}}(S^*_{\text{LAZY}})$ is not affected, it follows a contradiction with assumption that no workload unit could be processed earlier than in $OPT_{\text{e}}(I^*)$ without loosing optimality. Thus, we conclude that the total workload with release time strictly earlier than l_s^C is processed in $OPT_{\text{e}}(I^*)$ by time u_{s-1}^C.

Let us denote as B_s the processing block of S^*_{LAZY} in which workload that is processed in the optimal schedule $OPT_{\text{e}}(I^*)$ in C_s is processed. B_s is well defined because by definition of $LAZY$ all jobs processed by in schedule $OPT_{\text{e}}(I^*)$ in the processing block C_s are processed by $DEFER$ in only one processing block.

Let us define schedule $OPT_{\text{e}}(I^*)_c$ as follows. Let δ denote the set of workload units processed in $OPT_{\text{e}}(I^*)$ in B_s with release time strictly prior to l_s^C. Then let $OPT_{\text{e}}(I^*)_c$ be equal to $OPT_{\text{e}}(I^*)$ but if $\delta \neq \emptyset$, then process workload units in δ starting at time $\min\{l_s^B, l_s^C\}$. Rearrange the processing block C_{s-1} in case some idle periods were created by the interchange of workload units in set δ. So, energy consumed by schedule $OPT_{\text{e}}(I^*)_c$, $N_{\text{energy}}(OPT_{\text{e}}(I^*)_c)$, is equal to $N_{\text{energy}}(OPT_{\text{e}}(I^*))$. From definition of schedule S^*_{LAZY} and $OPT_{\text{e}}(I^*)_c$ it follows that:

$$\frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*))} = \frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*)_c)}.$$

Let us denote as E^1_{LAZY} the total energy consumed in S_{LAZY} from time 0 to time u_{s-1}^B. Let us denote as E^2_{LAZY} the total energy consumed in S^*_{LAZY} from time u_{s-1}^B until completion time. Let us denote as E^1_{OPT} and E^2_{OPT} the total energy consumed in $OPT_{\text{e}}(I^*)_c$ from time 0 to time u_{s-1}^C and the total energy consumed in $OPT_{\text{e}}(I^*)_c$ from time u_{s-1}^C to completion time respectively. Then it follows from these definitions and latter equality that:

$$\frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*))} \leq \frac{E^1_{LAZY} + E^2_{LAZY}}{E^1_{OPT} + E^2_{OPT}}.$$

Now, let us define as instance I_1 the instance formed by the jobs in instance I^* with release time at most u_{s-1}^B and I_2 the instance formed by the jobs in instance I^* with release time at least u_{s-1}^B. In I_2 release times and deadlines are the release times and deadlines in I^* minus u_{s-1}^B. From definitions it follows that $N_{\text{energy}}(S_{\text{LAZY}\,i}) \geq E^i_{LAZY}$ for $i = 1, 2$. And $N_{\text{energy}}(OPT_{\text{e}}(I_i)) \leq E^i_{OPT}$ for $i = 1, 2$. Then it follows that:

$$\begin{aligned}\frac{E^1_{LAZY}+E^2_{LAZY}}{E^1_{OPT}+E^2_{OPT}} &\leq \frac{\sum_{i=1}^{2} N_{\text{energy}}(S_{\text{LAZY}\,i})}{\sum_{i=1}^{2} N_{\text{energy}}(OPT_{\text{e}}(I_i))} \\ &\leq \max_{i=1,2} \frac{N_{\text{energy}}(S_{\text{LAZY}\,i})}{N_{\text{energy}}(OPT_{\text{e}}(I_i))}.\end{aligned}$$

We should assume WLOG that the latter inequality holds with equality otherwise it follows a contradiction with optimality of I^*. Given that the instance in

$$\arg\max_{i=1,2} \frac{N_{\text{energy}}(S_{\text{LAZY}\,i})}{N_{\text{energy}}(OPT_{\text{e}}(I_i))}$$

has strictly less workload than I^* then it follows a contradiction with the assumption that among instances in $\arg\max_I \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))}$ such that no job could be processed earlier I^* has the smallest total workload. Thus we can conclude that $OPT_{\text{e}}(I^*)$ has no "short" idle period.

□

Now we proceed to expose an instance, $\hat{I}$, with at most 3 idle periods, for which competitive ratio of algorithm DEFER is at most three times the competitive ratio of DEFER for instance I^*, $\frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*))}$. Let us define for an arbitrary processing block of schedule $OPT_{\text{e}}(I^*)$, C_i, the ordered set S_i as the set of time intervals in $\left[u^C_{i-1}, u^C_i\right]$, in which S^*_{LAZY} is idle and such that S_i has minimum cardinality. Let us denote as I^i_j the j^{th} element of set S_i. In other words, S_i is the set of disjoint idle periods of S^*_{LAZY} in time interval $\left[u^C_{i-1}, u^C_i\right]$ ordered in increasing order of their infimum.

Also let us define $\Omega(C_i)$ as the total workload processed by $DEFER$ in time interval $\left[u^C_{i-1}, u^C_i\right]$ if $0 < i < |OPT_{\text{e}}(I^*)|$, as the total workload processed by S^*_{LAZY} in time interval $[0, u^C_0]$ if $i = 0$ and as the total workload processed by $DEFER$ in time interval $[u^C_{i-1}, \infty)$ if $i = |S^*_{\text{LAZY}}|$. Using these definitions, the energy consumed by $OPT_{\text{e}}(I^*)$ and S^*_{LAZY} can be written as:

$$\begin{aligned}N_{\text{energy}}(OPT_{\text{e}}(I^*)) &= H \cdot \sum_{i=0}^{|OPT_{\text{e}}(I^*)|} \Omega(C_i) + \sum_{i=1}^{|OPT_{\text{e}}(I^*)|} N_{\text{energy}}(OPT_{\text{e}}(I_i)) \\ &= \sum_{i=0}^{|OPT_{\text{e}}(I^*)|} \left[H \cdot \Omega(C_i) + N_{\text{energy}}(OPT_{\text{e}}(I_i))\right]\end{aligned}$$

and

$$\begin{aligned} N_{\text{energy}}(S^*_{\text{LAZY}}) &= H \cdot \sum_{i=0}^{|OPT_{\text{e}}(I^*)|} \Omega\left(C_i\right) + \sum_{i=1}^{|OPT_{\text{e}}(I^*)|} \sum_{j=1}^{|S_i|} N_{\text{energy}}(I_i^j) \\ &= \sum_{i=0}^{|OPT_{\text{e}}(I^*)|} \left[H \cdot \Omega\left(C_i\right) + \sum_{j=1}^{|S_i|} N_{\text{energy}}(I_i^j) \right] \end{aligned}$$

respectively.

Let us define for $OPT_{\text{e}}(I^*)$:

$$C_{i^*} = \arg\max_{C_i} \frac{H \cdot \Omega\left(C_i\right) + \sum_{j=1}^{|S_i|} N_{\text{energy}}(I_i^j)}{H \cdot \Omega\left(C_i\right) + N_{\text{energy}}(OPT_{\text{e}}(I_i))}.$$

From definition of C_{i^*} and $N_{\text{energy}}(S^*_{\text{LAZY}}) \geq N_{\text{energy}}(OPT_{\text{e}}(I^*))$, it follows that:

$$\frac{E_{ALG}(I^*)}{E_{OPT}(I^*)} \leq \frac{H \cdot \Omega\left(C_{i^*}\right) + \sum_{j=1}^{|S_{i^*}|} N_{\text{energy}}(I_{i^*}^j)}{H \cdot \Omega\left(C_{i^*}\right) + N_{\text{energy}}(OPT_{\text{e}}(I_{i^*}))}.$$

Let us define $NC_{i^*} = H \cdot \Omega\left(C_{i^*}\right) + \sum_{j=1}^{|S_{i^*}|} N_{\text{energy}}(I_{i^*}^j)$ and $DC_{i^*} = H \cdot \Omega\left(C_{i^*}\right) + N_{\text{energy}}(OPT_{\text{e}}(I_{i^*}))$. Next we will expose an instance $\hat{I}$ such that the ratio between $\frac{NC_{i^*}}{DC_{i^*}}$ and the competitive ratio of realization $\hat{I}$ is bounded above by 3:

$$\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}} \leq 3$$

which together with:

$$\frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(I))} \leq \frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*))} \leq \frac{NC_{i^*}}{DC_{i^*}}$$

will imply that:

$$\frac{N_{\text{energy}}(S^*_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I^*))} \leq 3 \cdot \frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))},$$

which is the desire result.

Now we proceed to built an instance $\hat{I}$ and argue that:

$$\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}} \leq 3.$$

Let us consider the following cases:

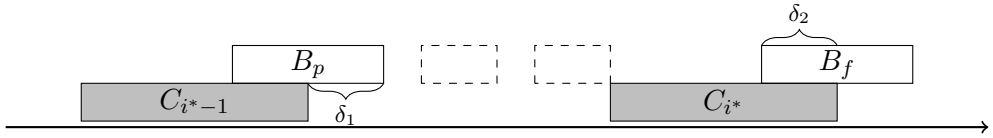


Figure 4.4.3: Case 1. There is a processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is a processing block that starts strictly before and finish strictly after $u^C_{i^*}$

1. *There is a processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is a processing block that starts strictly before and finish strictly after $u^C_{i^*}$.*

   Let us denote as B_p and B_f the processing block in S^*_{LAZY} that starts strictly before and finish strictly after $u^C_{i^*-1}$ and that starts strictly before and finish strictly after $u^C_{i^*}$ respectively. Let us also denote as δ_1 the total workload processed in B_p after time $u^C_{i^*-1}$ and as δ_2 the total workload processed in B_f before time $u^C_{i^*}$, see Figure 4.4.3. Let $J_{i^*} = \{j \in I^* : u^B_p < r_j < u^B_{f-1}\}$ Let $\hat{I}$ be the following realization:

   - Release at times $0, \ldots, \delta_1$, unitary jobs with deadlines $1, \ldots, \delta_1 + 1$ respectively.
   - For each job j in J_{i^*} release at time $r_j - u^C_{i^*-1}$ a job with workload ω_j and deadline $d_j - u^C_{i^*-1}$.
   - Release at time $u^C_{i^*} - \delta_2 - u^C_{i^*-1}$, δ_2 unitary jobs with span $s_{\max}$ and announce that no other job is going to be released in this realization.

   From definition of $\hat{I}$ it follows that $N_{\text{energy}}(S_{\hat{\text{LAZY}}}) \geq NC_{i^*}$ and $N_{\text{energy}}(OPT_{\text{e}}(\hat{I})) \leq DC_{i^*} + U$. This latter follows from the fact that $OPT_{\text{e}}(\hat{I})$ has at most 2 idle periods, the energy consumed by one of them is accounted in DC_{i^*} and U is an upper bound in the energy consumed during the second one. Then:

$$\begin{aligned}
\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{N_{\text{energy}}(S_{\hat{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}} &\leq \frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{NC_{i^*}}{DC_{i^*}+U}} \\
&\leq \frac{DC_{i^*} + U}{DC_{i^*}} \\
&\leq \frac{2 \cdot U + 2 \cdot H}{U + 2 \cdot H} \\
&\leq 2.
\end{aligned}$$

The third inequality follows from the fact that the idle period preceding C_{i^*} is a long idle period and from assumption that $\delta_1 > 0$ and $\delta_2 > 0$ it follows that $\Omega(C_{i^*}) \geq 2$. Then $DC_{i^*} \geq U + 2 \cdot H$.

2. *There is a processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is No processing block that starts strictly before and finish strictly after $u^C_{i^*}$.*

   Let us denote as B_p the processing block in S^*_{LAZY} that starts strictly before and finish

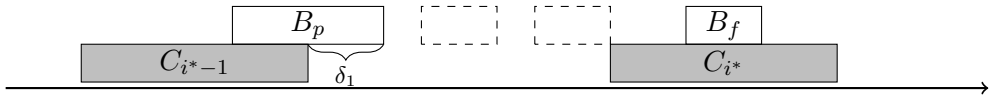


Figure 4.4.4: Case 2. here is a processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is No processing block that starts strictly before and finish strictly after $u^C_{i^*}$

strictly after $u^C_{i^*-1}$ and let us denote as B_f the processing block in $S_{ALG}(I^*)$ that finish the latest before $u^C_{i^*}$. Let us also denote as δ_1 the total workload processed in B_p after time $u^C_{i^*-1}$, see Figure 4.4.4. Let $J_{i^*} = \{j \in I^* : u^B_p < r_j \leq u^B_f\}$ Let $\hat{I}$ be the following realization:

- Release at times $0, \ldots, \delta_1$, unitary jobs with deadlines $1, \ldots, \delta_1 + 1$ respectively.
- For each job j in J_{i^*} at time $r_j - u^C_{i^*-1}$ with workload ω_j and deadline $d_j - u^C_{i^*-1}$.
- If J_{i^*} is not empty, release at time $\max_{j \in J_{i^*}} d_j - u^C_{i^*-1}$ a unitary job with span $s_{\max}$. Otherwise release at time $u^C_{i^*} - 1 - u^C_{i^*-1}$ a unitary job with span $s_{\max}$ and announce that is the last job of the realization.

From definition of $\hat{I}$, it follows that $N_{\text{energy}}(\hat{S_{\text{LAZY}}}) \geq NC_{i^*} + H$ and $N_{\text{energy}}(OPT_{\text{e}}(\hat{I})) \leq DC_{i^*} + H + U$. Thus, $\frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))} \geq \frac{NC_{i^*}+H}{DC_{i^*}+H+U}$. Then:

$$\begin{aligned}
\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{N_{\text{energy}}(\hat{S_{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}} &\leq \frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{NC_{i^*}+H}{DC_{i^*}+U+H}} \\
&\leq \frac{DC_{i^*} + H + U}{DC_{i^*}} \\
&\leq \frac{U + 2 \cdot H + U}{U + H} \\
&\leq 2
\end{aligned}$$

The third inequality follows because the idle period preceding block C_{i^*} is a long idle period. Additionally given that $\delta_1 > 0$ it holds that $DC_{i^*} \geq U + H$.

3. *There is no processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is No processing block that starts strictly before and finish strictly after $u^C_{i^*}$.*

   Let B_p be the processing block in S^*_{LAZY} that finish the latest before time $u^C_{i^*-1}$ and let

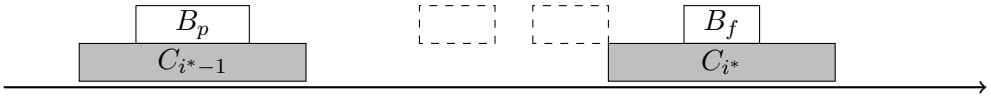


Figure 4.4.5: Case 3. There is No processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is No processing block that starts strictly before and finish strictly after $u^C_{i^*}$

B_f be the processing block in S^*_{LAZY} that finish the latest before $u^C_{i^*}$, see Figure 4.4.5. Let $J_{i^*} = \{j \in I^* : u^B_p < r_j \leq u^B_f\}$. Let $\hat{I}$ be the following realization:

- For each job j in J_{i^*} release a job at time $r_j - \min_{k \in J_{i^*}} r_k + 1$ with workload ω_j and deadline $d_j - \min_{k \in J_{i^*}} r_k + 1$.
- If set J_{i^*} is not empty, release at time $\max_{j \in J_{i^*}} d_j - \min_{k \in J_{i^*}} r_k + 1$ a unitary job with span $s_{\max}$. Otherwise release at time $l^C_{i^*} - \min_{k \in J_{i^*}} r_k + 1$ a unitary job with span 1. In both cases announce that such job is the las job in the realization.

From definition of $\hat{I}$, $N_{\text{energy}}(\hat{S_{\text{LAZY}}}) \geq NC_{i^*} + H$ and $N_{\text{energy}}(OPT_{\text{e}}(\hat{I})) \leq DC_{i^*} + 2 \cdot U + H$. This latter inequality follows because $OPT_{\text{e}}(\hat{I})$ has at most 3 idle periods. The energy of one of them is already accounted in DC_{i^*} and the energy of the other 2 is upper bounded by $2 \cdot U$. Then it follows that:

$$\begin{aligned}
\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{E_{ALG}(\hat{I})}{E_{OPT}(\hat{I})}} &\leq \frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{NC_{i^*}+H}{DC_{i^*}+U+H}} \\
&\leq \frac{DC_{i^*} + 2 \cdot U + H}{DC_{i^*} + H} \\
&\leq \frac{U + 2 \cdot U}{U} \\
&\leq 2.
\end{aligned}$$

The third inequality follows from the fact that idle period preceding C_{i^*} is long and then $DC_{i^*} \geq U$.

4. *There is no processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is a processing block that starts strictly before and finish strictly after $u^C_{i^*}$.* Let us denote as B_p the

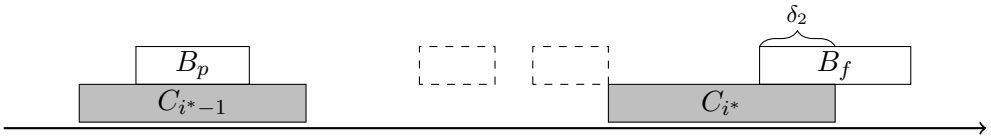


Figure 4.4.6: Case 4. There is No processing block that starts before and finish strictly after $u^C_{i^*-1}$, and there is a processing block that starts strictly before and finish strictly after $u^C_{i^*}$

processing block in $S_{\hat{\text{LAZY}}}$ with finishes the latest before time $u^C_{i^*-1}$. Let us denote as B_f the processing block in S^*_{LAZY} that starts strictly before and finish strictly after $u^C_{i^*}$ respectively. Let us denote as δ_2 the total workload processed in B_f before time $u^C_{i^*}$, see Figure 4.4.6. Let $J_{i^*} = \{j \in I^* : u^B_p < r_j \leq u^B_{f-1}\}$ Let $\hat{I}$ be the following realization:

- For each job j in J_{i^*} release a job at time $r_j - \min_{k \in J_{i^*}} r_k + 1$ with workload ω_j and deadline $d_j - \min_{k \in J_{i^*}} r_k + 1$.
- Release at time $u^C_{i^*} - \delta_2 - 1 - \min_{k \in J_{i^*}} r_k + 1$, δ_2 unitary jobs with span s_{max}. Announce that no other job is going to be released in such realization.

From definition of $\hat{I}$, $N_{\text{energy}}(S_{\hat{\text{LAZY}}}) = NC_{i^*}$ and $N_{\text{energy}}(OPT_{\text{e}}(\hat{I})) \leq DC_{i^*} + 2 \cdot U$. This latter inequality follows because $OPT_{\text{e}}(\hat{I})$ has at most 3 idle periods. The energy of one of them is already accounted in DC_{i^*} and the energy of the other 2 is upper bounded by $2 \cdot U$. Then it follows that:

$$\begin{aligned}
\frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{N_{\text{energy}}(S_{\hat{\text{LAZY}}})}{N_{\text{energy}}(OPT_{\text{e}}(\hat{I}))}} &\leq \frac{\frac{NC_{i^*}}{DC_{i^*}}}{\frac{NC_{i^*}}{DC_{i^*}+2 \cdot U}} \\
&\leq \frac{DC_{i^*} + 2 \cdot U}{DC_{i^*}} \\
&\leq \frac{U + H + 2 \cdot U}{U + H} \\
&= \frac{3 \cdot U + H}{U + H}
\end{aligned}$$

$$\begin{aligned} &= \frac{U+H}{U+3\cdot H}+\frac{2\cdot U}{U+H} \\ &\leq 3. \end{aligned}$$

The third inequality follows from the fact that $DC_{i^*} \geq U + H$.

□

Theorem 4.4.2. *If S_I is the set of instances of a constrained adversary with parameters $s_{\max}$ and $\rho_{\max} = 1$, such that for an arbitrary $I \in S_I$ $OPT_{\mathrm{e}}(I)$ has only one idle period. Then:*

$$\max_{I\in S_I}\frac{N_{\mathrm{energy}}(S_{\mathrm{LAZY}})}{N_{\mathrm{energy}}(OPT_{\mathrm{e}}(I))} = O\left(s_{\max}^{1/3}\right).$$

Proof. Let I be an arbitrary instance in $\arg\max_{I\in S_I} \frac{N_{\mathrm{energy}}(S_{\mathrm{LAZY}})}{N_{\mathrm{energy}}(OPT_{\mathrm{e}}(I))}$. Let n_1 be the number of processing blocks in S_{LAZY} whose intersection with the unique idle period of the optimal schedule is non empty. From Claim 2 in Proposition 4.4.1 it follows that there exist an optimal schedule $OPT_{\mathrm{e}}(I)$ with only long idle periods.

Now we proceed to determine the minimum workload processed in each processing block $B_1, \ldots, B_{n_1}$ to guarantee that the idle period has length at most $U + 1$.

Let B_i be an arbitrary processing block such that $i = 1, \ldots, n_1$. Let us define $Q(t, i) = \sum_{j\in A(t)} \omega_j$ for an arbitrary $i \in A(t)$ and let $\rho_{\mathrm{e}}(t) = \sum_{j\in A(t)} \rho_j$. From definition of $LAZY$ it follows that:

$$l_i^B \geq d_j^i - \left\lfloor (Q(t_i, j_i) - 1)\cdot\left(\frac{1}{\rho_{\mathrm{e}}(t_i)} - 1\right)\right\rfloor - Q(t_i, j_i)$$

For all $t_i \in (u_{i-1}^B, l_i^B]$ and some $j_i \in A(t_i)$ with deadline d_j^i. We may assume from now onwards that $t_i = l_i^B$. From assumption that the optimal schedule has only one idle period together with definition of B_i, it follows that jobs processed by $LAZY$ in B_i are processed in $OPT_{\mathrm{e}}(I)$ in C_1, and thus:

$$d_j^i \geq u_{i-1}^B + \sum_{k=i}^{n_1} \left[U + 1 + Q(t_k, j_k)\right]. \tag{4.4.3}$$

Additionally from the maximum span constraint, it follows that the time elapsed between the first release time of the instance and l_1^C is at most s_{max}. We can assume WLOG that the first job is released at time 1. Let I_i be the length of the idle period preceding processing block B_i:

$$\begin{aligned}
I_i &= l_i^B - u_{i-1}^B \\
&\geq d_j^i - \left\lfloor (Q(t_i, j_n) \cdot \left(\frac{1}{\rho_e(t_i)} - 1\right) \right\rfloor - Q(t_i, j_i) - u_{i-1}^B \\
&\geq d_j^i - (Q(t_i, j_i) - 1) \cdot \left(\frac{1}{\rho_e(t_i)} - 1\right) - Q(t_i, j_i) - u_{i-1}^B \\
&= d_j^i - \frac{Q(t_i, j_i) - 1}{\rho_e(t_i)} - 1 - u_{i-1}^B \\
&\geq d_j^i - \frac{Q(t_i, j_i) - 1}{\frac{Q(t_i, j_i)}{d_j^i - u_{i-1}^B}} - 1 - u_{i-1}^B \\
&= \left(d_j^i - u_{i-1}^B\right) \cdot \left[1 - \frac{Q(t_i, j_i) - 1}{Q(t_i, j_i)}\right] - 1. \qquad (4.4.4)
\end{aligned}$$

The Inequality (4.4.4) follows because $\rho_e(t_n) \geq \frac{Q(t_n, j_n)}{d_j^n - u_{n-1}^B}$. Now we proceed to find a lower bound in the amount of workload processed at each processing block B_i, $i = 1, \ldots, n_1$ using the Inequalities (4.4.3) and (4.4.4),

$$\begin{aligned}
I_i &= \left(d_j^i - u_{i-1}^B\right) \cdot \left[1 - \frac{Q(t_i, j_i) - 1}{Q(t_i, j_i)}\right] - 1 \\
&\geq \left(\sum_{k=i}^{n_1} (U + 1 + Q(t_k, j_k))\right) \cdot \left[1 - \frac{Q(t_i, j_i) - 1}{Q(t_i, j_i)}\right] - 1 \\
&= \frac{\sum_{k=i}^{n_1} (U + 1 + Q(t_k, j_k))}{Q(t_i, j_i)} - 1
\end{aligned}$$

Then it follows that:

$$Q(t_i, j_i) \geq \frac{\sum_{k=i+1}^{n_1} [U + 1 + Q(t_k, j_k)]}{I_i}, \quad i < n_1 \qquad (4.4.5)$$

$$Q(t_{n_1}, j_{n_1}) \geq \frac{U + 1}{I_{n_1}}. \qquad (4.4.6)$$

Additionally, we can assume WLOG that $I_i \leq U + 1$ because $LAZY$ manages power during idle periods in such a way that an interval larger than $U+1$ cannot improve the relative performance of OPT respect to $LAZY$. After some manipulations of Inequality (4.4.5) and (4.4.6) together with the assumption $I_i \leq U + 1$ for $i = 1, \ldots, n_1$ we obtain:

$$Q(t_{n_1 - i}, j_{n_1 - i}) \geq i + \frac{\sum_{k=1}^{i} k}{U + 1}, i = 1, \ldots n_1 - 1$$

$$Q(t_{n_1}, j_{n_1}) \geq 1.$$

From the maximum span constraint, the span of any job is at most $s_{\max}$, it follows that from the time of the first release to l_1^C there are at most $s_{\max} - 1$ time units. Then it follows that:

$$\begin{aligned} s_{\max} &\geq \sum_{i=1}^{n_1} [U + 1 + Q(t_{n-i}, j_{n-i})] \\ &\geq n_1 \cdot (U+1) + 1 + \sum_{i=1}^{n_1} \left[i + \frac{\sum_{k=1}^{i} k}{U+1} \right] \\ &\geq n_1 \cdot (U+1) + 1 + \frac{n_1 \cdot (n_1+1)}{2} + \frac{1}{U+1} \cdot \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6} \end{aligned}$$

Then we can conclude that for a given $s_{\max}$ a feasible U and n_1 most satisfy the following relation:

$$n_1 \cdot (U+1)^2 + \left(1 + \frac{n_1 \cdot (n_1+1)}{2} - s_{\max}\right) \cdot (U+1) + \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6} \leq 0$$

Which is equivalent to:

$$U + 1 \leq \frac{\left(s_{\max} - \frac{n_1 \cdot (n_1+1)}{2} - 1\right) + \sqrt{\left(s_{\max} - \frac{(n_1+1) \cdot n_1}{2} - 1\right)^2 - 4 \cdot n_1 \cdot \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6}}}{2 \cdot n_1}. \quad (4.4.7)$$

Let us denote the RHS of the last inequality as $l(n, s_{\max})$

$$l(n, s_{\max}) = \frac{\left(s_{\max} - \frac{n_1 \cdot (n_1+1)}{2} - 1\right) + \sqrt{\left(s_{\max} - \frac{(n_1+1) \cdot n_1}{2} - 1\right)^2 - 4 \cdot n_1 \cdot \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6}}}{2 \cdot n_1},$$

and let $f(n_1, U)$ be:

$$f(n_1, U) = \frac{2 \cdot n_1 \cdot U + \left[1 + \frac{n_1 \cdot (n_1+1)}{2} + \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6\,(U+1)}\right]}{U + \left[1 + \frac{n_1 \cdot (n_1+1)}{2} + \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6\,(U+1)}\right]}.$$

From the assumption that the optimal schedule has only one idle period which has length at least $U + 1$ it follows that:

$$\begin{aligned} \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))} &\leq \frac{n_1 \cdot 2 \cdot U + \sum_{i=1}^{n_1} Q(t_{n_1-i}, j_{n_1-i})}{U + \sum_{i=1}^{n_1} Q(t_{n_1-i}, j_{n_1-i})} + 4 \\ &\leq \frac{2 \cdot n_1 \cdot U + \left[1 + \frac{n_1 \cdot (n_1+1)}{2} + \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6\,(U+1)}\right]}{U + \left[1 + \frac{n_1 \cdot (n_1+1)}{2} + \frac{n_1 \cdot (n_1+1) \cdot (n_1+2)}{6\,(U+1)}\right]} + 4 \\ &= \frac{12\,U^2\,n + 3\,U\,n^2 + 15\,U\,n + 6\,U + n^3 + 6\,n^2 + 5\,n + 6}{6\,U^2 + 3\,U\,n^2 + 3\,U\,n + 12\,U + n^3 + 6\,n^2 + 5\,n + 6} + 4. \end{aligned}$$

$$= \quad f(n_1, U) + 4.$$

We proceed to find an upper bound on the competitive ratio for a given $s_{\max}$ by maximizing $f(n_1, U)$ subject to feasibility constraint impose by the constraint $U + 1 \leq l(n_1, s_{\max})$. The feasibility constraint comes from the Inequality 4.4.7. Let consider the following optimization problem, Problem 7, with variables n_1 and U:

Problem 7.

$$\begin{aligned} \underset{n_1, U}{\textit{maximize}} \quad & f(n_1, U) \\ \textit{subject to} \quad & \\ & l(n_1, s_{\max}) \qquad \geq U + 1. \\ & U \geq 0, n_1 \geq 1 \end{aligned}$$

To solve this optimization problem we first argue that $\frac{\delta f(n_1, U)}{\delta U}$ is non negative for any value of $U \geq 0$ and $n_1 \geq 1$.

$$\begin{aligned} \frac{\delta f(n_1, U)}{\delta U} &= \frac{6 \cdot (2\,n - 1) \cdot \left(3\,U^2\,n^2 + 3\,U^2\,n + 6\,U^2 + 2\,U\,n^3 + 12\,U\,n^2 + 10\,U\,n + 12\,U + n^3 + 6\,n^2 + 5\,n + 6\right)}{\left((6\,U^2 + 3\,U\,n^2 + 3\,U\,n + 12\,U + n^3 + 6\,n^2 + 5\,n + 6\right)^2} \\ &\geq 0, \end{aligned}$$

for any $U > 0$, $n \geq 1$. It follows that there exist an optimal solution of optimization problem 7 such that the feasibility constraint holds with equality. This holds because for any feasible n, setting U to $l(n, s_{\max}) - 1$ which is non-negative by definition, implies a non-negative improvement in the objective value. Then optimization Problem 7 is equivalent to unidimensional optimization problem:

Problem 8.

$$\underset{n \geq 1}{\textit{maximize}} \quad f(n, l(n, s_{\max}) - 1)$$

From definition of $f(n, U)$, $f(n, l(n, s_{\max}) - 1)$ could be written as:

$$\begin{aligned} f(n, l(n, s_{\max}) - 1) &= \frac{2 \cdot n \cdot U(n) + Q(n)}{U(n) + Q(n)} \\ &= \frac{(2 \cdot n - 1) \cdot U(n) + U(n) + Q(n)}{U(n) + Q(n)} \\ &= \frac{(2 \cdot n - 1) \cdot U(n)}{U(n) + Q(n)} + 1 \end{aligned}$$

$$\begin{aligned} &= \frac{2\cdot n-1}{1+\frac{Q(n)}{U(n)}}+1 \\ &\leq \frac{2\cdot n}{1+\frac{Q(n)}{U(n)}}+1, \end{aligned}$$

in which $Q(n)$ and $U(n)$ are defined as follow:

$$\begin{aligned} Q(n) &= 1+\frac{n\cdot(n+1)}{2}+\frac{n\cdot(n+1)\cdot(n+2)}{6\,(U+1)} \\ U(n) &= \frac{\left(s_{\max}-\frac{n\cdot(n+1)}{2}-1\right)+\sqrt{\left(s_{\max}-\frac{(n+1)\cdot n}{2}-1\right)^2-4\cdot n\cdot\frac{n\cdot(n+1)\cdot(n+2)}{6}}}{2\cdot n}-1. \end{aligned}$$

Let us define $h(n)=\frac{2\cdot n}{1+\frac{Q(n)}{U(n)}}$ and $g(n)=1+\frac{Q(n)}{U(n)}$. In Proposition D.0.1 in Appendix D we argued that when $s_{\max}\to\infty$, $\Theta\left(s_{max}^{1/3}\right)\in\arg\min_n h(n)$ which implies the desired result.

□

Corollary 4.4.3. *If I is an arbitrary realization of a constrained adversary with parameters $s_{\max}$ and $\rho_{\max}\leq 1$. Then:*

$$\max_I \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))} = O\left(s_{max}^{1/3}\right)$$

Proof. From Theorem 4.4.1 it follows that there exist an instance $\hat{I}$ such that the competitive ratio of instance I is at most 3 times the competitive ratio of instance $\hat{I}$. Additionally $\hat{I}$ has at most 3 idle periods such that $\hat{S_{\text{LAZY}}}$ processes workload in at most one of them. Thus there is at most one idle period such that the ratio between the energy consumed by S_{LAZY} during its idle periods that intersect with the idle period of $OPT_{\text{e}}(\hat{I})$ under consideration could be larger than 2, denote such idle period as I_2. The ratio between the energy consumed during I_2 in $\hat{S_{\text{LAZY}}}$ and the energy consumed in $OPT_{\text{e}}(\hat{I})$ during the same time interval is upper bounded by the upper bound on the competitive ratio found among instances in which there exist an optimal schedule with only one idle period. This follows because in the latter case the adversary is less constraint, thus it is more powerful than in $\hat{I}$ during idle period I_2 of $OPT_{\text{e}}(\hat{I})$, thus

$$\max_I \frac{N_{\text{energy}}(S_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I))} \leq \max_{\{I':\exists\ OPT_{\text{e}}(I')\text{ s.t }|OPT_{\text{e}}(I')|=1\}} \frac{N_{\text{energy}}(S'_{\text{LAZY}})}{N_{\text{energy}}(OPT_{\text{e}}(I'))}.$$

This together with the result in Theorem 4.4.2 imply the desired result. □

4.5 Discussion

We presented an online algorithm for the minimum energy scheduling problem when the processor has two power states and consumes a fixed amount of energy when it transits between power states. We considered two versions of this problem. We perform a competitive analysis subject to a constraint oblivious adversary. In all the cases that we consider we provide bounds as a function of the upper bound on the span of the jobs that can be released by the adversary, $s_{\max}$. First we study the problem in which energy minimization is equivalent to minimizing the total number of idle periods. We show that the competitive ratio of an arbitrary online algorithm is at least $\left\lfloor \frac{s_{\max}}{3} \right\rfloor + 2$, for $s_{\max} \geq 5$, 2 for $s_{\max} \in \{3, 4\}$ and LAZY is trivially optimal for $s_{\max} \in \{1, 2\}$. We provide an algorithm with competitive ratio upper bounded by $\frac{s_{\max}}{2} + 2$. Lastly we consider the problem in which the total energy consumption is a function of the length of each idle period in the resultant schedule. In this case we provide asymptotic bounds. We show that the competitive ratio of an arbitrary online algorithm is lower bounded by $\Theta(\ln s_{\max})$. We also provide an upper bound on the competitive ratio of our algorithm, we establish that the competitive ratio of DEFER is upper bounded by $\Theta\left(s_{\max}^{1/3}\right)$.

Many questions remain to be answered. Even though we provide an efficient algorithm when the workload is not necessary integral, our algorithm when the workload is integral does not run in polynomial time and the bounds that we provide hold when the workload is integral. Is there an efficient online algorithm with a provable better competitive ratio than our algorithm? We provide asymptotic bounds in $s_{\max}$, however remains unknown bounds when $s_{\max}$ is not large.

Bibliography

[1] D.J. Abraham, A. Blum, and T. Sandholm. Clearing algorithms for barter exchange markets: enabling nationwide kidney exchanges. In *Proceedings of the 8th ACM conference on Electronic commerce*, pages 295–304. ACM, 2007.

[2] Susanne Albers. Energy-efficient algorithms. *Communications of the ACM*, 53(5):86–96, 2010.

[3] I. Ashlagi, D. Gamarnik, M.A. Rees, and A.E. Roth. The need for (long) chains in kidney exchange. Technical report, National Bureau of Economic Research, 2012.

[4] I. Ashlagi and A.E. Roth. Free riding and participation in large scale, multi-hospital kidney exchange. Technical report, Working paper, 2011.

[5] I. Ashlagi and A.E. Roth. Individual rationality and participation in large scale, multi-hospital kidney exchange. Technical report, National Bureau of Economic Research, 2011.

[6] Itai Ashlagi, Felix Fischer, Ian Kash, and Ariel D Procaccia. Mix and match. In *Proceedings of the 11th ACM conference on Electronic commerce*, pages 305–314. ACM, 2010.

[7] Itai Ashlagi, Felix Fischer, Ian A Kash, and Ariel D Procaccia. Mix and match: A strategyproof mechanism for multi-hospital kidney exchange. 2011.

[8] Itai Ashlagi, Felix Fischer, Ian A Kash, and Ariel D Procaccia. Mix and match: A strategyproof mechanism for multi-hospital kidney exchange. *Games and Economic Behavior*, 2013.

[9] J. Augustine, S. Irani, and C. Swamy. Optimal power-down strategies. *SIAM Journal on Computing*, 37(5):1499–1516, 2008.

[10] Pranjal Awasthi and Tuomas Sandholm. Online stochastic optimization in the large: Application to kidney exchange. In *IJCAI*, volume 9, pages 405–411, 2009.

[11] P. Baptiste. Scheduling unit tasks to minimize the number of idle periods: a polynomial time algorithm for offline dynamic power management. In *Proceedings of the seventeenth annual ACM-SIAM symposium on Discrete algorithm*, pages 364–367. ACM, 2006.

[12] P. Baptiste, M. Chrobak, and C. Dürr. Polynomial time algorithms for minimum energy scheduling. *Algorithms–ESA 2007*, pages 136–150, 2007.

[13] Luiz André Barroso. The price of performance. *Queue*, 3(7):48–53, 2005.

[14] Allan Borodin, Jon Kleinberg, Prabhakar Raghavan, Madhu Sudan, and David P Williamson. Adversarial queuing theory. *Journal of the ACM (JACM)*, 48(1):13–38, 2001.

[15] Ioannis Caragiannis, Aris Filos-Ratsikas, and Ariel Procaccia. An improved 2-agent kidney exchange mechanism. *Internet and Network Economics*, pages 37–48, 2011.

[16] John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Optimizing kidney exchange with transplant chains: Theory and reality. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 711–718. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[17] John P Dickerson, Ariel D Procaccia, and Tuomas Sandholm. Failure-aware kidney exchange. In *Proceedings of the fourteenth ACM conference on Electronic commerce*, pages 323–340. ACM, 2013.

[18] J.P. Dickerson, A.D. Procaccia, and T. Sandholm. Dynamic matching via weighted myopia with application to kidney exchange. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence*, 2012.

[19] Ringo Doe. Optn / srtr annual report: Transplant data 1999-2008, 2009.

[20] S.G Hanson and T.S. Bentley. 2011 u.s organ and tissue transplant cost estimates and discussion, 2011.

[21] Dorit S Hochbaum. *Approximation algorithms for NP-hard problems*. PWS Publishing Co., 1996.

[22] S. Irani, R. Gupta, and S. Shukla. Competitive analysis of dynamic power management strategies for systems with multiple power savings states. In *Proceedings of the conference on Design, automation and test in Europe*, page 117. IEEE Computer Society, 2002.

[23] S. Irani, S. Shukla, and R. Gupta. Online strategies for dynamic power management in systems with multiple power-saving states. *ACM Transactions on Embedded Computing Systems (TECS)*, 2(3):325–346, 2003.

[24] Sandy Irani and Kirk R Pruhs. Algorithmic problems in power management. *ACM SIGACT News*, 36(2):63–76, 2005.

[25] Anna R Karlin, Mark S Manasse, Lyle A McGeoch, and Susan Owicki. Competitive randomized algorithms for nonuniform problems. *Algorithmica*, 11(6):542–571, 1994.

[26] Jonathan Koomey. Growth in data center electricity use 2005 to 2010. *The New York Times*, 49(3), 2011.

[27] D. Manlove and G. OMalley. Paired and altruistic kidney donation in the uk: algorithms and experimentation. *Experimental Algorithms*, pages 271–282, 2012.

[28] Herwig-Ulf Meier-Kriesche and Bruce Kaplan. Waiting time on dialysis as the strongest modifiable risk factor for renal transplant outcomes: A paired donor kidney analysis1. *Transplantation*, 74(10):1377–1381, 2002.

[29] ML Melcher, DB Leeser, HA Gritsch, J Milner, S Kapur, S Busque, JP Roberts, S Katznelson, W Bry, H Yang, et al. Chain transplantation: Initial experience of a large multicenter program. *American Journal of Transplantation*, 12(9):2429–2436, 2012.

[30] National Kidney Registry. Nkr - paired exchange results quarterly report, 2013.

[31] A. Roth, T. Sonmez, and U. Unver. Efficient kidney exchange: Coincidence of wants in markets with compatibility-based preferences. *American Economic Review*, 97(3), 2007.

[32] Alvin E Roth, Tayfun Sonmez, and M Utku Unver. Kidney exchange. Technical report, National Bureau of Economic Research, 2003.

[33] Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. Kidney exchange. *The Quarterly Journal of Economics*, 119(2):457–488, 2004.

[34] Alvin E Roth, Tayfun Sonmez, and M Utku Unver. Efficient kidney exchange: Coincidence of wants in a structured market. Technical report, National Bureau of Economic Research, 2005.

[35] Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. A kidney exchange clearinghouse in new england. *American Economic Review*, pages 376–380, 2005.

[36] Alvin E Roth, Tayfun Sönmez, and M Utku Ünver. Pairwise kidney exchange. *Journal of Economic Theory*, 125(2):151–188, 2005.

[37] Tayfun Sönmez and MU Ünver. Market design for kidney exchange. *Oxford Handbook of Market Design, Oxford University Press (to appear)*, 2010.

[38] Panagiotis Toulis and David C Parkes. A random graph model of kidney exchanges: efficiency, individual-rationality and incentives. In *Proceedings of the 12th ACM conference on Electronic commerce*, pages 323–332. ACM, 2011.

[39] M.U. Ünver. Dynamic kidney exchange. *The Review of Economic Studies*, 77(1):372–414, 2010.

[40] C.B. Wallis, K.P. Samy, A.E. Roth, and M.A. Rees. Kidney paired donation. *Nephrology Dialysis Transplantation*, 26(7):2091–2099, 2011.

[41] Robert A Wolfe, Valarie B Ashby, Edgar L Milford, Akinlolu O Ojo, Robert E Ettenger, Lawrence YC Agodoa, Philip J Held, and Friedrich K Port. Comparison of mortality in all patients on dialysis, patients on dialysis awaiting transplantation, and recipients of a first cadaveric transplant. *New England Journal of Medicine*, 341(23):1725–1730, 1999.

[42] Frances Yao, Alan Demers, and Scott Shenker. A scheduling model for reduced cpu energy. In *Foundations of Computer Science, 1995. Proceedings., 36th Annual Symposium on*, pages 374–382. IEEE, 1995.

Appendix A

Example mechanism

BALANCE-AND-AUGMENT

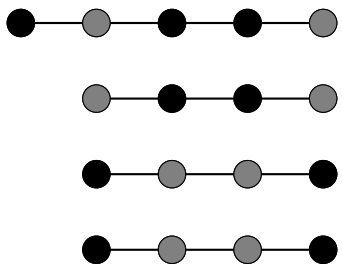


Figure A.0.1: compatibility graph G. Notice that graph G is a disconnected graph with 4 components.

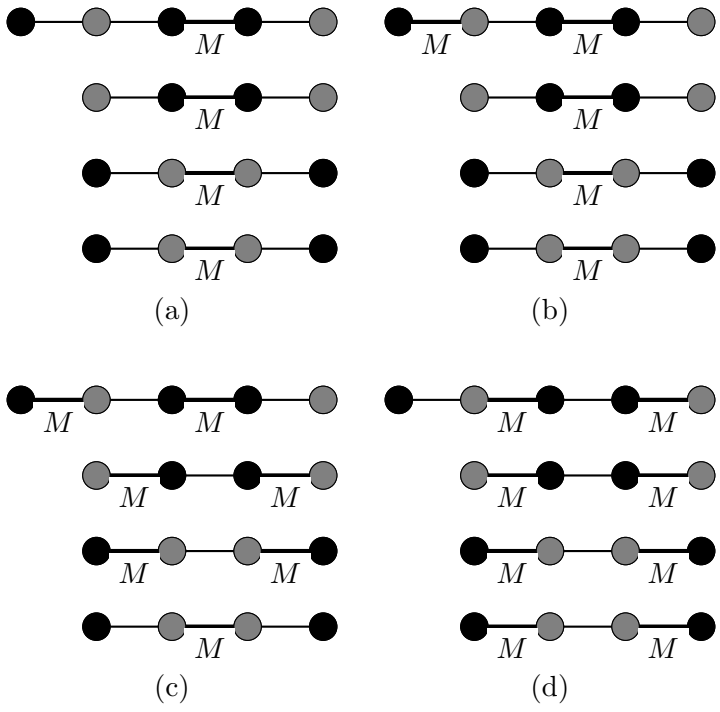


Figure A.0.2: Output of mechanism BALANCE when applied on G at the end of each step of the algorithm. (a) step 1 (b) step 2 (c) step 3 (d) step 4.

Appendix B

BALANCE-ALL and BALANCE-ALL-AND-AUGMENT: strategy-proofness and bounds

Proposition B.0.6. *Let $f(G)$ be the output of the algorithm* `BALANCE-ALL` *on the graph G. Then either $f(G)$ is a maximum-cardinality matching, or it maximizes the number of matched nodes for at least one of the agents. In other words, $f(G)$ maximizes the total score, or the gray score or the black score.*

Proof. Let M be the matching at the end of step 3 of the mechanism `BALANCE-ALL` (i.e., just before step 4 is executed). We already know that every M-augmenting path is of one color, and without loss of generality, we take this to be black. The algorithm terminates after executing step 4 as many times as needed, updating M in each of those steps. Note that for the algorithm to terminate, either $U_R = \emptyset$; or $V_B = \emptyset$; or the matching M has no black augmenting paths. We show that in the last case, the algorithm finds a max-cardinality matching, whereas in the other two cases the algorithm maximizes the number of gray nodes that are matched. If $U_R = \emptyset$, every gray node is matched, so the gray score is maximized in $M := f(G)$ trivially. Similarly, if there is no black augmenting path with respect to M, then there is no augmenting path with respect to M (as no augmenting path involving a free gray node is created in the last step), and so M is a max-cardinality matching.

The only remaining case to consider is when $V_B = \emptyset$, but $U_R \neq \emptyset$. We wish to argue that M

maximizes the gray score. Let M' be a matching that maximizes the gray score (and if there are many such matchings, pick one with maximum cardinality, breaking ties arbitrarily). We argue that the gray score in M is exactly the same as that in M'. Consider the components of the symmetric difference of M and M'. It is well known that each component in the symmetric difference is either an even-length cycle, or a path (of odd or even length).

- In an even-length cycle both M and M' match all the nodes, and hence the same number of gray nodes.
- In an even-length path, we claim that both leaf nodes must belong to the same agent. If not, there is a gray leaf node and a black one; if the gray leaf node is not matched by M, there is an even-length alternating path from a gray node to a black node, contradicting $V_B = \emptyset$; if the gray leaf node is matched by M, then flipping the edges of M and M' only in this component, we can find a matching M'' that matches more gray nodes than M' does, contradicting the definition of M'. And when both leaf nodes belong to the same agent, both M and M' match the same number of gray nodes.
- In an odd-length path, if the first and last edges are matched in M, both M and M' match the same number of gray nodes (as the leaf nodes of the path have to be black, otherwise it contradicts definition of matching M'). If the first and last edges are matched in M', then the two leaf nodes of the path have to be black as well: otherwise there is an M-augmenting path from a gray node, which we assume does not exist. Again, M and M' match the same number of gray nodes.

Thus we find that M matches the same number of gray nodes as M'. As M' maximizes the gray score, so does M. □

Proposition B.0.7. *Mechanism* `BALANCE-ALL` *is strategyproof.*

Proof. Let G be an arbitrary graph and let $f(G)$ be the output of mechanism `BALANCE-ALL` for graph G. Let R_e be the maximum gray score in the graph (over all matchings); similarly, let B_e be the maximum black score. For any matching M, we let $\mu_R(M)$ and $\mu_B(M)$ be, respectively, the number of gray and black nodes matched in M. For an arbitrary graph F, we denote as $\mu(F)$ the number of nodes matched in a maximum cardinality matching of graph F. Recall that R_i (resp. B_i)

denotes the maximum number of nodes matched by the gray (resp. black) agent in the subgraph induced by the gray (resp. black) nodes alone. Proposition B.0.6 implies that the outcome of BALANCE-ALL is a maximum-cardinality matching, or one that maximizes the gray score, or one that maximizes the black score. We prove that BALANCE-ALL is strategyproof in each case.

Suppose BALANCE-ALL finds a maximum cardinality matching $f(G)$ on the graph G. Then $\mu_R(f(G)) = R_i + \Delta$ and $\mu_B(f(G)) = B_i + \Delta$, for some $\Delta \leq \min\{R_e - R_i, B_e - B_i, \frac{\mu(G)-R_i-B_i}{2}\}$. Suppose the black agent "hides" a subset of his nodes so that H is the subgraph induced by the hidden nodes (known only to the black agent), and let $G' = \{G_k\}$ the set of graphs seen by the mechanism. For the black agent to do strictly better with this strategy, the gray agent must do strictly worse as BALANCE-ALL found a max-cardinality matching on G. Thus, we must have $\mu_R(f(\{G^k\})) < \mu_R(f(G))$. But by the key property of BALANCE-ALL, $\mu_R(f(\{G^k\})) = (\sum_k R^k) + \Delta' = R_i + \Delta'$. This implies that $\Delta' < \Delta$. Now, the number of black nodes matched when black hides is simply $\mu(H) + \mu_B(f(G'))$, and we have:

$$\begin{aligned} \mu(H) + \mu_B(f(G')) &= \mu(H) + \sum_k B_i^k + \Delta' \\ &\leq B_i + \Delta' < B_i + \Delta = \mu_B(f(G)). \end{aligned}$$

Thus, it is not possible for the black agent to do better by hiding any portion of his subgraph in this case. A similar argument applies for the gray agent.

Suppose BALANCE-ALL maximizes the score of one of the agents. Without loss of generality, we assume that it maximizes the score of the gray agent, and so $\mu_R(f(G)) = R_e$. Clearly, the gray agent has no incentive to manipulate in this case. Again, suppose the black agent hides a subset of his nodes so that H is the induced subgraph, and let $\{G_k\}$ the set of graphs seen by the mechanism. Then, the black score in the "manipulated" graph is upper-bounded by

$$\mu(H) + \sum_k B_i^k + (R_e - R_i).$$

To see why, note that the additional utility over the utility that each agent can achieve on their own on the manipulated graph G', Δ', is equal to $\min\left\{R'_e - R_i, B'_e - \sum_k B_i^k, \frac{\mu(G')-R_i-\sum_k B_i^k}{2}\right\}$, which is upper bounded by $R_e - R_i$, because $R'_e \leq R_e$. The latter implies that $\Delta' < R_e - R_i$. This together with the fact that $\mu(H) + \sum_k B_i^k \leq B_i$ imply that the black agent does not benefit from hiding any of its nodes.

□

Proposition B.0.8. *The Mechanism* `BALANCE-ALL-AND-AUGMENT` *can be implemented in polynomial time, has an approximation ratio of 4/3, and is truthful in expectation.*

Proof. That `BALANCE-ALL-AND-AUGMENT` can be implemented in polynomial time is straightforward. The expected number of nodes matched by `BALANCE-ALL-AND-AUGMENT` is at least 1/2 OPT with probability 1/2, and the output of `BALANCE`, which is a maximal matching, with probability 1/2. Simplifying, we find that this mechanism matches at least 3/4 OPT, proving the approximation ratio. Note that this mechanism returns a maximal matching, and by a result of Caragiannis et al. [15], this is best possible among mechanisms that always return a maximal matching. The only property we need to prove is that the mechanism is truthful in expectation.

It follows clearly that an agent that has an expected utility equal to the maximum utility that it could achieves in G has no interest in be untruthful, thus we may assume that if an agent has incentive to be untruthful, it has an expected utility strictly less than the best that its maximum possible utility in graph G. Assume WLOG that agent black has incentive to hide some of its nodes.

We shall show that when the black agent hides a subset of nodes, the expected gain from this manuever in step 2 is offset by an equal or larger loss in step 1. As the matchings computed in these steps are equally likely to be chosen, the result follows. As before, suppose black hides a set of nodes such that H is the subgraph induced by these hidden black nodes (known only to the black agent) and $G' = \{G^k\}$ is the graph induced by all remaining nodes in G. Let M be the output of `BALANCE-ALL` on G and let M' the output of `BALANCE-ALL` on G' together with a maximum cardinality matching of H; let OPT be the max-cardinality matchings obtained augmenting all the augmenting paths respect to M and let OPT' be the matching obtained from augmenting all augmenting paths respect to the output of `BALANCE-ALL` on G' together with the maximum cardinality matching of H.

Define $\Delta_{BR} := \min\{R_e - R_i, B_e - B_i, \mu(\text{OPT}) - \frac{B_i+R_i}{2}\}$. The definition of `BALANCE-ALL` implies that $\mu_B(M) = B_i + \Delta_{BR}$: the last term in the definition of Δ_{BR} accounts for the case when `BALANCE-ALL` finds a max-cardinality matching; in that case, exactly as many additional gray

nodes are matched as the number of additional black nodes, and the total score should be $\mu(\text{OPT})$, so the gain of $\mu(\text{OPT}) - (B_i + R_i)$ is shared equally by both agents. Similarly, we can define Δ'_{BR} as the number of gray nodes matched by `BALANCE` in G' beyond $R_i(B'_i)$.

Consider the gray score in the graph G'. This is given by $R_i + \Delta'_{BR}$. But we also know that or the gray score in M is the maximum possible or M is a maximum cardinality matching of G. In both cases, it follows that $\Delta_{BR} \geq \Delta'_{BR}$. Thus, $\Delta_{BR} = \Delta'_{BR} + \delta$ for some $\delta \geq 0$. We shall show that:

$$\mu_B(\text{OPT'}) \quad \leq \quad \mu_B(\text{OPT}) + \delta, \tag{B.0.1}$$

and that

$$\delta + \mu_B(M') \quad \leq \quad \mu_B(M). \tag{B.0.2}$$

Adding these inequalities, we find $\mu_B(\text{OPT'}) + \mu_B(M') \leq \mu_B(\text{OPT}) + \mu_B(M)$, which implies the result. Inequality (B.0.2) follows trivially from definition of M and M':

$$\begin{aligned} \mu_B(M) - \mu_B(M') &= B_i + \Delta_{BR} - \sum_k B_i^k - \mu(H) - \Delta'_{BR} \\ &\geq B_i + \Delta'_{BR} + \delta - B_i - \Delta'_{BR} \\ &= \delta \end{aligned}$$

Now we are left to argue inequality (B.0.1). We first argue that $\mu_R(\text{OPT}) - \mu_R(\text{OPT'}) \leq \delta$. Observe that $\mu_R(M') = R_i + \Delta'_{BR} = R_i + \Delta_{BR} - \delta$; using $\mu_R(M) = R_i + \Delta_{BR}$, we see that $\mu_R(M) - \mu_R(M') = \delta$. But we already know that $\mu_R(\text{OPT}) = \mu_R(M)$, and $\mu_R(\text{OPT'}) \geq \mu_R(M')$. Putting all this together, we have

$$\mu_R(\text{OPT}) - \mu_R(\text{OPT'}) \;\leq\; \mu_R(M) - \mu_R(M') \;=\; \delta. \tag{B.0.3}$$

Note that $\mu_R(\text{OPT}) + \mu_B(\text{OPT})$ is the size of the max-cardinality matching in the original graph, and $\mu_R(\text{OPT'}) + \mu_B(\text{OPT'})$ is the size of a matching of G obtained when black matches some of his nodes internally, and the rest of the graph is matched optimally. Trivially, we have:

$$\mu_R(\text{OPT'}) + \mu_B(\text{OPT'}) \;\leq\; \mu_R(\text{OPT}) + \mu_B(\text{OPT}),$$

which when rearranged gives

$$\mu_B(\text{OPT'}) - \mu_B(\text{OPT}) \;\leq\; \mu_R(\text{OPT}) - \mu_R(\text{OPT'}) \;\leq\; \delta,$$

where the last inequality follows from inequality (B.0.3). □

Appendix C

Parameters estimation: technical report

The data source is publicly available in the the Organ procurement and transplantation network (OPTN) website[1]. "OPTN is the unified transplant network established by the United States Congress under the National Organ Transplant Act (NOTA) of 1984."

C.1 Forecasting the UNO's waiting list arrival in the next 10 years

For each blood type we use the time series of the yearly number of patients that arrive by first time to the UNO's waiting list to forecast the number of patients that will arrive in the next 10 years (our time study horizon) to the UNO's waiting list. This will allow us to forecast the proportion of each ABO blood type in the stream of arrivals in our time study horizon.
Let $\{x_t^p(b)\}$ denote the time series of yearly patient's additions to the UNO's waiting list for blood type $b \in \{O, A, B, AB\}$. Figure C.1.1 shows the time plot of the number of patients of each ABO blood type that joined the UNO's waiting list by first time from 1995 to 2012. From the time plots in figure C.1.1 we observe that the time series corresponding to each blood type waiting list's new additions have a trend and so they are not stationary.

In Figure C.2.2 we plot the time series corresponding to $x_t^p(b) - x_{t-1}^p(b)$ and their autocorrelation (ACF) and partial autocorrelation function (PACF). The resulting ACF and PACF reveal uncorre-

[1] http://optn.transplant.hrsa.gov/optn/

lated time series, allowing us to conclude that the time series $x_t^p(b) - x_{t-1}^p(b)$ for $b \in \{O, A, B, AB\}$ can be modeled as a constant plus an unrelated error term. In a more general sense, it suggests that $AR(1)$, $x_t^p(b) - \mu(b) = \alpha(b) \cdot \left(x_{t-1}^p(b) - \mu(b)\right) + \epsilon_t$ in which $\epsilon \sim \mathcal{N}(m, \sigma^2)$, is a good candidate to model $x_t^p(b)$ for every $b \in \{O, A, B, AB\}$. Next we estimate the parameters $\alpha(b)$ and $\mu(b)$ for

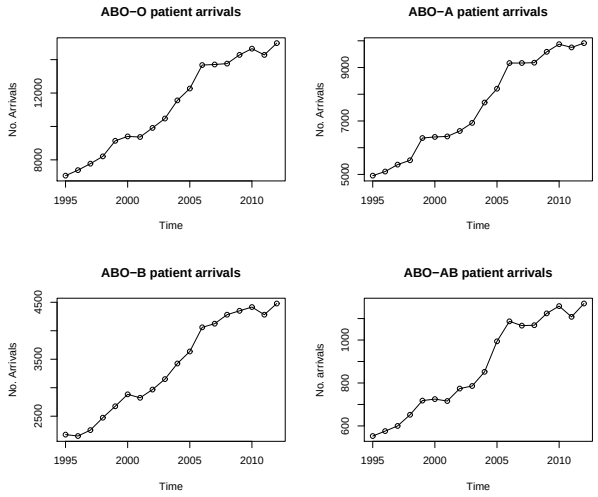


Figure C.1.1: Time plots of time series $x_t^p(b)$ for $b \in \{O, A, B, AB\}$

$b \in \{O, A, B, AB\}$ and analyze the residuals. The analysis of the residuals include test of normality (Shapiro-Wilk test) and independence (Box-Ljung test). For all ABO blood types the null hypothesis of normality can not be rejected with a level of significance of 1%. A similar result hold for the independence test, for all ABO blood type the null hypothesis of independence can not be rejected with a level of significants of 1%. See Table C.1 for the details.

Then we conclude that the time series of the new yearly additions to the UNO's waiting list for each ABO blood type can be modeled as an AR(1) time series. We finally use these models to forecast the number of patients of each ABO blood type that will arrives to the UNO's waiting list in the next 10 years (study time horizon), see Table C.2. We use as forecasting method Kalman filter through the built-in *predict* procedure in R.

Parameters estimation		**Shapiro-Wilk test**	**Box-Ljung test**
$\hat{\alpha}(O) = 0.99$	$\hat{\mu}(O) = 11050$	W = 1	X-squared = 0.04
$\sigma(\hat{\alpha}(O)) = 0.02$	$\sigma(\hat{\mu}(O)) = 3530$	p-value = 0.9983	df = 1, p-value = 0.8331
$\hat{\alpha}(A) = 0.98$	$\hat{\mu}(A) = 7452$	W = 0.9	X-squared = 0.3
$\sigma(\hat{\alpha}(A)) = 0.02$	$\sigma(\hat{\mu}(A)) = 2185$	p-value = 0.4245	df = 1, p-value = 0.6047
$\hat{\alpha}(B) = 0.98$	$\hat{\mu}(B) = 3331$	W = 0.4214	X-squared = 2
$\sigma(\hat{\alpha}(B)) = 0.02$	$\sigma(\hat{\mu}(B)) = 1025$	p-value = 0.9	df = 1, p-value = 0.1308
$\hat{\alpha}(AB) = 0.98$	$\hat{\mu}(AB) = 863$	W = 1	X-squared = 2
$\sigma(\hat{\alpha}(AB)) = 0.03$	$\sigma(\hat{\mu}(AB)) = 264$	p-value = 0.945	df = 1, p-value = 0.1934

Table C.1: AR(1) estimated parameters of times series $x_t^p(p)$ for $p \in \{O, A, B, AB\}$ and the corresponding analysis of residuals (normality and independence).

C.2 Forecasting the UNO's living donors arrivals in the next 10 years

We denote as $\left\{y_t^d(b)\right\}$ the time series of the yearly number of living donors of blood type $b \in \{O, A, B, AB\}$ that have donated to the UNO's waiting list form 1989 to 2012. Figure C.2.1 shows the time plots of times series $\left\{y_t^d(b)\right\}$ for $b \in \{O, A, B, AB\}$. Time plots in Figure C.2.1 suggest that for all the ABO blood types the time series of living donors that donated to the UNO's waiting list is not stationary and present a trend. In Figure C.2.3 we plot the time series corresponding to $y_t^d(b) - y_{t-1}^d(b)$ and their ACF and PACF. The resulting ACF and PACF reveal uncorrelated time series, suggesting that the time series $y_t^d(b) - y_{t-1}^d(b)$ for $b \in \{O, A, B, AB\}$ can be modeled as a constant plus an unrelated error term. In a more general sense, it suggests that $AR(1)$, $y_t^d(b) - \upsilon(b) = \beta(b) \cdot \left(y_{t-1}^d(b) - \upsilon(b)\right) + \epsilon_t$ in which $\epsilon \sim \mathcal{N}(m, \sigma^2)$, is a good candidate to model $y_t^d(b)$ for every $b \in \{O, A, B, AB\}$. Next we estimate the parameters $\beta(b)$ and $\upsilon(b)$ for $b \in \{O, A, B, AB\}$ and analyze the residuals. The analysis of the residuals include test of normality (Shapiro-Wilk test) and independence (Ljung-Box test). For all ABO blood types the null hypothesis of normality can not be rejected with a level of significance of 5%. A similar result holds for the independence test, for all ABO blood type the null hypothesis of independence can not be rejected with a level of significants of 5%. See Table C.3 for the details. Then we conclude that the time series of the

ABO	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
$\hat{x}_t^p(O)$	14933	14878	14824	14770	14718	14666	14614	14564	14514	14465
$\sigma\left(\hat{x}_t^p(O)\right)$	627	881	1071	1228	1364	1484	1591	1690	1780	1863
$\hat{x}_t^p(A)$	9873	9833	9795	9756	9719	9682	9646	9610	9575	9540
$\sigma\left(\hat{x}_t^p(A)\right)$	419	588	715	819	908	987	1057	1121	1180	1234
$\hat{x}_t^p(B)$	4461	4444	4428	4411	4395	4380	4364	4349	4334	4319
$\sigma\left(\hat{x}_t^p(B)\right)$	187	262	319	365	405	441	473	502	528	553
$\hat{x}_t^p(AB)$	1164	1158	1152	1146	1140	1135	1129	1124	1119	1113
$\sigma\left(\hat{x}_t^p(AB)\right)$	57	80	97	111	123	133	143	151	159	166

Table C.2: Predicted number of arrivals of each ABO blood type in 10 years time horizon

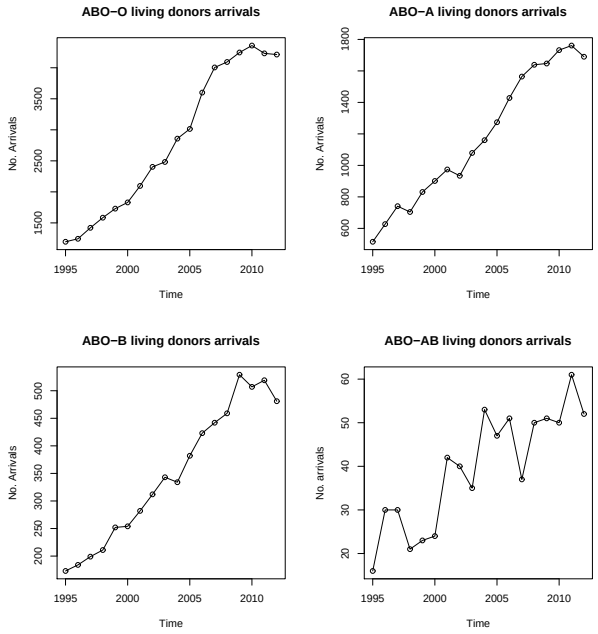


Figure C.2.1: Time plots of time series $y_t^d(b)$ for $b \in \{O, A, B, AB\}$

number of living donors that donated to the UNO's waiting list for each ABO blood type can be modeled as an AR(1) time series. We finally use these models to forecast the number of donors of each ABO blood type that will arrives to the UNO's waiting list in the next 10 years (study time horizon), see Table C.4. We use as forecasting method Kalman filter through the built-in *predict* procedure in R.

Parameters estimation		**Shapiro-Wilk test**	**Ljung-Box test**
$\hat{\alpha}(O) = 0.99$	$\hat{\mu}(O) = 2717$	W = 1	X-squared = 3
$\sigma\left(\hat{\alpha}(O)\right) = 0.02$	$\sigma\left(\hat{\mu}(O)\right) = 1362$	p-value = 0.6754	df = 1, p-value = 0.1126
$\hat{\alpha}(A) = 0.98$	$\hat{\mu}(A) = 1112$	W = 0.9	X-squared = 1
$\sigma\left(\hat{\alpha}(A)\right) = 0.02$	$\sigma\left(\hat{\mu}(A)\right) = 522$	p-value = 0.07003	df = 1, p-value = 0.2579
$\hat{\alpha}(B) = 0.98$	$\hat{\mu}(B) = 331$	W = 1	X-squared = 1
$\sigma\left(\hat{\alpha}(B)\right) = 0.03$	$\sigma\left(\hat{\mu}(B)\right) = 132$	p-value = 0.9144	df = 1, p-value = 0.3158
$\hat{\alpha}(AB) = 0.7$	$\hat{\mu}(AB) = 38$	W = 1	X-squared = 3
$\sigma\left(\hat{\alpha}(AB)\right) = 0.2$	$\sigma\left(\hat{\mu}(AB)\right) = 7$	p-value = 0.4702	df = 1, p-value = 0.107

Table C.3: AR(1) estimated parameters of times series $y_t^d(p)$ for $p \in \{O, A, B, AB\}$ and the corresponding analysis of residuals (normality and independence).

ABO	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022
$\hat{x}_t^p(O)$	4192	4171	4150	4130	4110	4091	4071	4052	4033	4015
$\sigma\left(\hat{x}_t^p(O)\right)$	242	339	412	473	525	571	613	651	685	718
$\hat{x}_t^p(A)$	1681	1673	1664	1656	1648	1640	1632	1624	1616	1609
$\sigma\left(\hat{x}_t^p(A)\right)$	96	134	163	187	208	226	242	257	271	283
$\hat{x}_t^p(B)$	478	474	471	468	464	461	458	455	452	450
$\sigma\left(\hat{x}_t^p(B)\right)$	31	43	53	60	66	72	77	81	85	89
$\hat{x}_t^p(AB)$	48	46	44	42	41	41	40	40	39	39
$\sigma\left(\hat{x}_t^p(AB)\right)$	9	11	12	13	13	14	14	14	14	14

Table C.4: Predicted yearly number of living donors by ABO blood type that will be transplanted in 2013-2022 time horizon

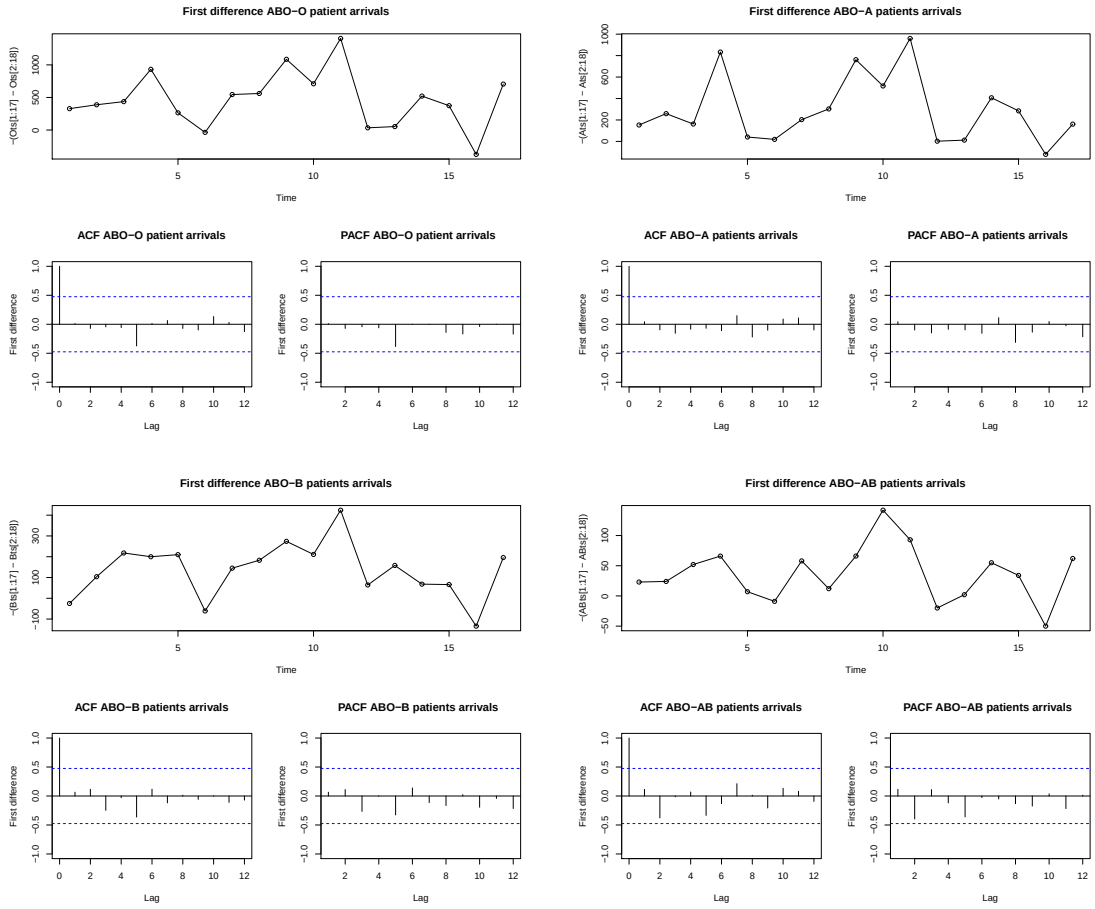


Figure C.2.2: Time plot, ACF and PACF of $x_t^p(b) - x_{t-1}^p(b)$ for $b \in \{O, A, B, AB\}$

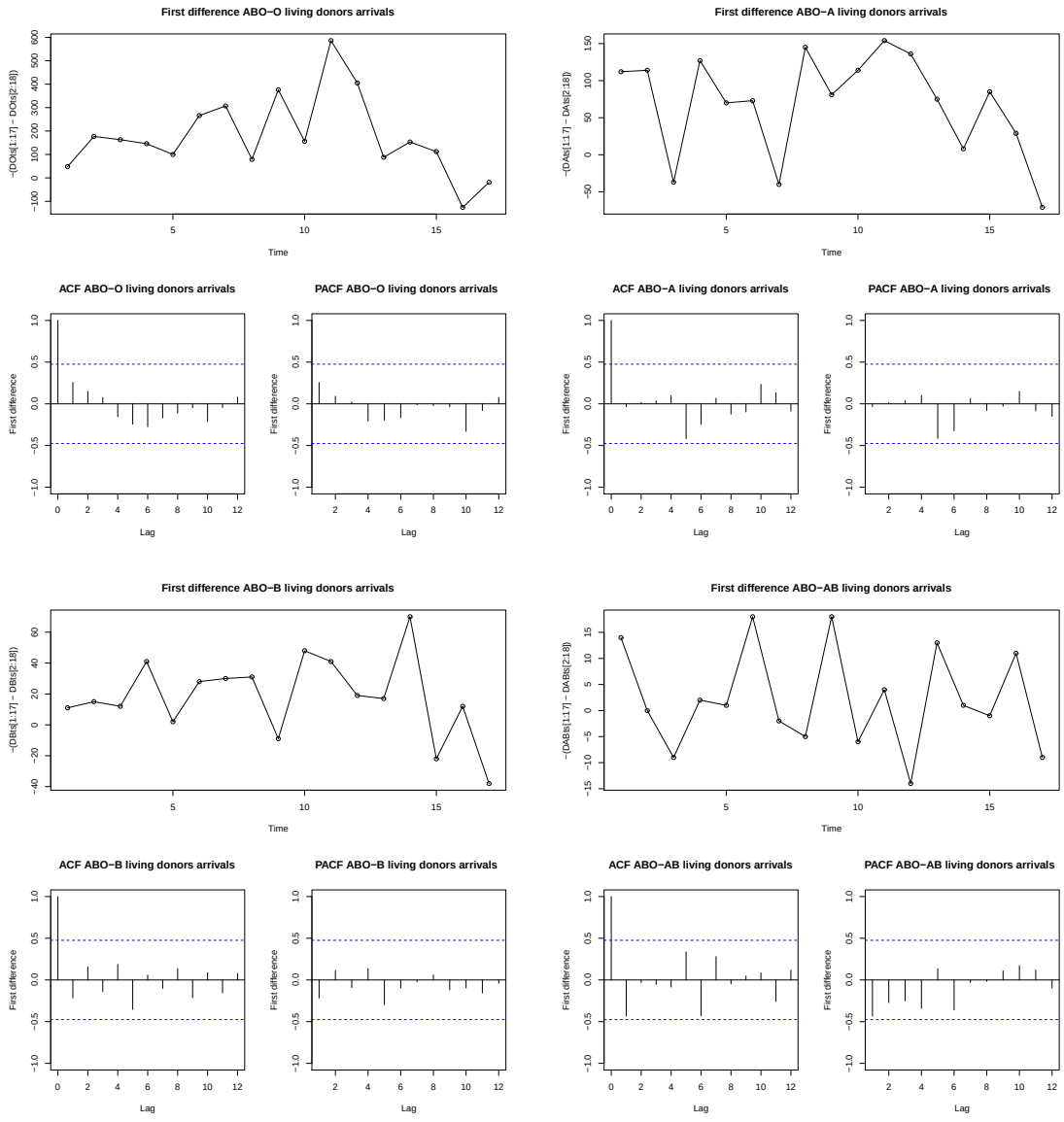


Figure C.2.3: Time plot, ACF and PACF of $y_t^d(b) - y_{t-1}^d(b)$ for $b \in \{O, A, B, AB\}$

Appendix D

Proposition used in Proof of Theorem 4.4.2

Proposition D.0.1. *If* $s_{\max} \to \infty$,

$$s_{\max}^{\frac{1}{3}} \in \arg\max_{n} \frac{2\,n}{1+\frac{Q(n)}{U(n)}},$$

for

$$\begin{aligned} Q(n) &= 1+\frac{n\cdot(n+1)}{2}+\frac{n\cdot(n+1)\cdot(n+2)}{6\,(U+1)} \\ U(n) &= \frac{\left(s_{\max}-\frac{n\cdot(n+1)}{2}-1\right)+\sqrt{\left(s_{\max}-\frac{(n+1)\cdot n}{2}-1\right)^2-4\cdot n\cdot\frac{n\cdot(n+1)\cdot(n+2)}{6}}}{2\cdot n}-1. \end{aligned}$$

Proof. We start arguing that $h'(1)$ is non negative, then we will argue that the numerator of $h'(n)$ is non-increasing and we will finish exposing that $h'(n)$ is negative for at least one value in the domain of $h(n)$. These will allow us to conclude that $h(n)$ has exactly one stationary point in all its domain. Then it will follow that $h(n)$ has a global maximizer.

From definition $h'(n)=\frac{2\cdot g(n)-2\cdot n\cdot g'(n)}{g(n)^2}$. Let us define the following functions:

$$Y(n,s_{\max}) = 12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\,(n+1)\,(n+2)$$

$$Z(n, s_{\max}) = n - 2\,\mathrm{s}_{\max} - \frac{\sqrt{3\,Y(n, s_{\max})}}{3} + n^2 + 2$$

$$W(n, s_{\max}) = 5\,n - 2\,\mathrm{s}_{\max} - \frac{\sqrt{3\,Y(n, s_{\max})}}{3} + n^2 + 2$$

$$V(n, s_{\max}) = \frac{n}{6} + \mathrm{s}_{\max} + 2\,n\,\mathrm{s}_{\max} + \frac{9\,n^2}{2} + \frac{5\,n^3}{3} - 1$$

From manipulations it follows that:

$$\begin{aligned} g(n) - n\,g'(n) &= \frac{2\,n^3}{W(n, s_{\max})} - \frac{4\,n}{W(n, s_{\max})} + \frac{32\,n^3}{W(n, s_{\max})\,Z(n, s_{\max})} \\ &- \frac{16\,n^5}{W(n, s_{\max})\,Z(n, s_{\max})} \\ &- \frac{16\,n^3 \left(\frac{\frac{n}{2} + \frac{\sqrt{3}\,V(n, s_{\max})}{2\sqrt{Y(n, s_{\max})}} + \frac{1}{4}}{n} - \frac{\frac{n}{4} - \frac{\mathrm{s}_{\max}}{2} - \frac{\sqrt{3\,Y(n, s_{\max})}}{12} + \frac{n^2}{4} + \frac{1}{2}}{n^2} \right) \left(\frac{n\,(n+1)}{2} - \frac{4\,n^2\,(n+1)\,(n+2)}{Z(n, s_{\max})} + 1 \right)}{W(n, s_{\max})^2} \\ &- \frac{32\,n^3\,(n+1)\,(n+2)}{W(n, s_{\max})\,Z(n, s_{\max})} + \frac{32\,n^4\,(n+1)\,(n+2)\left(n + \frac{\sqrt{3}\,V(n, s_{\max})}{\sqrt{Y(n, s_{\max})}} + \frac{1}{2} \right)}{W(n, s_{\max})\,Z^2(n, s_{\max})} + 1 \end{aligned}$$

Then:

$$\begin{aligned} g(1) - g'(1) &= \frac{1}{\mathrm{s}_{\max} + \sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} - 4} \\ &- \frac{132}{(\mathrm{s}_{\max} - 3)\,\sqrt{36\,(\mathrm{s}_{\max} - 2)^2 - 144} - 30\,\mathrm{s}_{\max} + 6\,{\mathrm{s}_{\max}}^2 + 24} \\ &+ \frac{144 \left(\frac{24}{2\,\mathrm{s}_{\max} + \frac{\sqrt{36\,(\mathrm{s}_{\max} - 2)^2 - 144}}{3} - 4} + 2 \right) \left(\frac{\mathrm{s}_{\max}}{2} + \frac{\sqrt{36\,(\mathrm{s}_{\max} - 2)^2 - 144}}{12} - \frac{\sqrt{3}\left(\frac{3\,\mathrm{s}_{\max}}{2} + \frac{8}{3} \right)}{\sqrt{12\,(\mathrm{s}_{\max} - 2)^2 - 48}} - \frac{3}{2} \right)}{\left(6\,\mathrm{s}_{\max} + \sqrt{36\,(\mathrm{s}_{\max} - 2)^2 - 144} - 24 \right)^2} \\ &- \frac{4\left(9\,\mathrm{s}_{\max} + 9\,\sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} + 16 \right)}{\sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)}\left(\mathrm{s}_{\max} + \sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} - 2 \right)^2 \left(\mathrm{s}_{\max} + \sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} - 4 \right)} \\ &+ 1 \end{aligned}$$

Which is positive for $s_{\max} = 10$ (≈ 0.2720) and which, as we will argue is non decreasing respect to $s_{\max}$, then it follows that $g(1) - g'(1)$ is positive for all $s_{\max} \geq 10$. To argue that $g(1) - g'(1)$ is increasing respect to $s_{\max}$ we expose its derivative, respect to $s_{\max}$, $[g(1) - g'(1)]'$, and argue that is positive for $s_{\max} = 5$, 42.7580. Additionally the numerator and denominator of $[g(1) - g'(1)]'$:

$$\begin{aligned} N\left\{ [g(1) - g'(1)]' \right\} &= 91392\,\mathrm{s}_{\max} + 28672\,\sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} + 13904\,(\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4))^{\frac{3}{2}} \\ &+ 3792\,(\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4))^{\frac{5}{2}} - 139264\,{\mathrm{s}_{\max}}^2 + 101168\,{\mathrm{s}_{\max}}^3 - 29664\,{\mathrm{s}_{\max}}^4 \\ &+ 2352\,{\mathrm{s}_{\max}}^5 + 144\,{\mathrm{s}_{\max}}^6 - 2176\,\mathrm{s}_{\max}\,\sqrt{\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4)} \\ &+ 3936\,\mathrm{s}_{\max}\,(\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4))^{\frac{3}{2}} + 144\,\mathrm{s}_{\max}\,(\mathrm{s}_{\max}\,(\mathrm{s}_{\max} - 4))^{\frac{5}{2}} - 16384 \end{aligned}$$

$$\begin{aligned} D\left\{[g(1)-g'(1)]'\right\} &= 21504\,{\mathrm{s_{max}}}^2 - 1200\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{5}{2}} - 240\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{7}{2}} \\ &\quad -768\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{3}{2}} - 59136\,{\mathrm{s_{max}}}^3 + 57792\,{\mathrm{s_{max}}}^4 - 27600\,{\mathrm{s_{max}}}^5 \\ &\quad +7008\,{\mathrm{s_{max}}}^6 - 912\,{\mathrm{s_{max}}}^7 + 48\,{\mathrm{s_{max}}}^8 + 192\,\mathrm{s_{max}}\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{3}{2}} \\ &\quad +288\,\mathrm{s_{max}}\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{5}{2}} + 48\,\mathrm{s_{max}}\left(\mathrm{s_{max}}\ (\mathrm{s_{max}}-4)\right)^{\frac{7}{2}} \end{aligned}$$

respectively, are increasing and given that both have values for $s_{\max} = 10$ positive, $3.4140e+008$, $8.9787e+008$ respectively. Then it follows that $[g(1)-g'(1)]'$ is positive for all $s_{\max} \geq 10$, then we can conclude that $g(1)-g'(1)$ is increasing in $s_{\max}$. Then $h'(1)$ is positive for all $s_{\max} \geq 10$.
To show that $g(n)-n\,g'(n)$ is decreasing. First we argue that $\frac{2\,n^3-4\,n}{W(n,s_{\max})}$ is decreasing. We will show that $W(1,s_{\max})$ is negative and $\frac{1}{W(n,s_{\max})}$ is decreasing, which imply that $\frac{1}{W(n,s_{\max})}$ is a decreasing function in n and multiplying by an increasing function, positive for all $n \geq 2$ results in a decreasing function.

$$\begin{aligned} W(1,s_{\max}) &= 8 - \frac{\sqrt{36\left(\mathrm{s_{max}}-2\right)^2-144}}{3} - 2\,\mathrm{s_{max}} \\ &< 0 \qquad \forall s_{\max} \geq 4 \end{aligned}$$

and the derivative of $\frac{1}{W(n,s_{\max})}$ respect to n is negative for all $n \geq 2$, $s_{\max} \geq 4$:

$$-\frac{9\left(2\,n+\frac{\sqrt{3}\left(6\,n+36\,\mathrm{s_{max}}+72\,n\,\mathrm{s_{max}}+162\,n^2+60\,n^3-36\right)}{18\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s_{max}}+1\right)^2-8\,n^2\,(n+1)\,(n+2)}}+5\right)}{\left(15\,n-6\,\mathrm{s_{max}}+3\,n^2-\sqrt{3}\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s_{max}}+1\right)^2-8\,n^2\ (n+1)\ (n+2)}+6\right)^2},$$

Which imply that $\frac{1}{W(n,s_{\max})}$ is negative for $n \geq 1$ and decreasing, then multiplying by $2\,n^3-4\,n$ just made this function decrease faster, for $n \geq 2$.

Now we will argue that $\frac{32\,n^3-16\,n^5-32n^3\,(n+1)\,(n+2)}{W(n,s_{\max})\,Z(n,s_{\max})}$ is also decreasing. To show this we will show that $\frac{1}{Z(n,s_{\max})\,W(n,s_{\max})}$ is increasing and start positive. Which implies that this latter function multiply by the decreasing function $32\,n^3-16\,n^5-32n^3\,(n+1)\,(n+2)$ that is negative for $n=1$ result in a decreasing function. Let us define the following functions:

$$A(n,s_{\max}) = 12\left(\frac{n\ (n+1)}{2}-\mathrm{s_{max}}+1\right)^2 - 8\,n^2\ (n+1)\ (n+2)$$

$$B(n,s_{\max}) = \frac{\sqrt{3}\ \left(6\,n+36\,\mathrm{s_{max}}+72\,n\,\mathrm{s_{max}}+162\,n^2+60\,n^3-36\right)}{18\sqrt{A(n,s_{\max})}}$$

Then first derivative of $\frac{1}{Z(n,s_{\max})\,W(n,s_{\max})}$ respect to n can be written as:

$$\frac{27\ (B(n,s_{\max})+2\,n+1)}{\left(3\,n-6\,\mathrm{s}_{\max}-\sqrt{3}\,\sqrt{A(n,s_{\max})}+3\,n^2+6\right)^2\ \left(-15\,n+6\,\mathrm{s}_{\max}+\sqrt{3}\,\sqrt{A(n,s_{\max})}-3\,n^2-6\right)}$$
$$+\frac{27\ (B(n,s_{\max})+2\,n+5)}{\left(-3\,n+6\,\mathrm{s}_{\max}+\sqrt{3}\,\sqrt{A(n,s_{\max})}-3\,n^2-6\right)\ \left(15\,n-6\,\mathrm{s}_{\max}-\sqrt{3}\,\sqrt{A(n,s_{\max})}+3\,n^2+6\right)^2}$$

We are left to argue that $\frac{1}{-15\cdot n+6\cdot s_{\max}+\sqrt{3\,A(n,s_{\max})}-3\,n^2-6}$ is increasing and is positive for $n=1$, $\frac{1}{6\,\mathrm{s}_{\max}+2\,\sqrt{3}\,\sqrt{3\,(\mathrm{s}_{\max}-2)^2-12}-24}$. Its derivative is given by:

$$\frac{6\,n+\frac{2\,\sqrt{3}\left(\frac{n}{2}+3\,\mathrm{s}_{\max}+6\,n\,\mathrm{s}_{\max}+\frac{27\,n^2}{2}+5\,n^3-3\right)}{\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\,(n+1)\,(n+2)}}+15}{\left(15\,n-6\,\mathrm{s}_{\max}+3\,n^2-\sqrt{3}\,\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\ (n+1)\ (n+2)}+6\right)^2}\quad\geq\quad 0,\quad\forall n\geq 1.$$

Now we will argue that $\frac{32\,n^4\,(n+1)\,(n+2)\left(n+\frac{\sqrt{3}\,V(n,s_{\max}))}{\sqrt{Y(n,s_{\max})}}\right)+\frac{1}{2}}{W(n,s_{\max})\,Z^2(n,s_{\max})}$ is decreasing. For that we will argue that $\frac{1}{Z^2(n,s_{\max})}$ is increasing and positive and that $n^4\,(n+1)\,(n+2)\ \left(n+\frac{\sqrt{3}\,V(n,s_{\max})}{\sqrt{Y(n,s_{\max})}}\right)$ is increasing and positive. Given that we previously argued that $\frac{1}{W(n,s_{\max})}$ is decreasing and negative, then it follows the desire result. The derivative of $\frac{1}{Z^2(n,s_{\max})}$ is given by:

$$\frac{54\ \left(2\,n+\frac{\sqrt{3}\left(6\,n+36\,\mathrm{s}_{\max}+72\,n\,\mathrm{s}_{\max}+162\,n^2+60\,n^3-36\right)}{18\,\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\,(n+1)\,(n+2)}}+1\right)}{\left(-3\,n+6\,\mathrm{s}_{\max}+3\,n^2+\sqrt{3}\,\sqrt{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2+8\,n^2\ (n+1)\ (n+2)}-6\right)^3}$$

To express the derivative of $n^4\,(n+1)\,(n+2)\ \left(n+\frac{\sqrt{3}\,V(n,s_{\max})}{\sqrt{Y(n,s_{\max})}}\right)$, let us define the function $D(n,s_{\max})$ and $F(n,s_{\max})$ as:

$$D(n,s_{\max}) \;=\; n+\sqrt{\frac{3}{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\ (n+1)\ (n+2)}}\left(\frac{n}{6}+\mathrm{s}_{\max}+2\,n\,\mathrm{s}_{\max}+\frac{9\,n^2}{2}+\frac{5\,n^3}{3}-1\right).$$

$$F(n,s_{\max}) \;=\; \frac{1}{12\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\ (n+1)\ (n+2)}$$

Then derivative can be written as:

$$n^4\ (n+1)\ D(n,s_{\max})+n^4\ (n+2)\ D(n,s_{\max})+4\,n^3\ (n+1)\ (n+2)\ D(n,s_{\max})$$

$$
\begin{array}{l}
+n^4\ (n+1)\ (n+2)\ \left(\sqrt{3}\,\sqrt{F(n,s_{\max})}\ \left(5\,n^2+9\,n+2\,\mathrm{s}_{\max}+\dfrac{1}{6}\right)\right.\\
+\dfrac{\sqrt{3}\,F(n,s_{\max})^{\frac{3}{2}}\ \left(\frac{n}{6}+\mathrm{s}_{\max}+2\,n\,\mathrm{s}_{\max}+\frac{9\,n^2}{2}+\frac{5\,n^3}{3}-1\right)}{2}\\
\left.\left(2\,n+12\,\mathrm{s}_{\max}+24\,n\,\mathrm{s}_{\max}+54\,n^2+20\,n^3-12\right)+1\right)
\end{array}
$$

Which is a positive function for $n \geq 1$. Finally we are going to argue that:

$$
-\frac{16\,n^3\ \left(\frac{\frac{n}{2}+\frac{\sqrt{3}\,V(n,s_{\max})}{2\,\sqrt{Y(n,s_{\max})}}+\frac{1}{4}}{n}-\frac{\frac{n}{4}-\frac{\mathrm{smax}}{2}-\frac{\sqrt{3\,Y(n,s_{\max})}}{12}+\frac{n^2}{4}+\frac{1}{2}}{n^2}\right)\ \left(\frac{n\,(n+1)}{2}-\frac{4\,n^2\,(n+1)\,(n+2)}{Z(n,s_{\max})}+1\right)}{W(n,s_{\max})^2}
$$

is decreasing in n. From the fact that $1/W(n,s_{\max})$ is negative for $n \geq 1$ and decreasing, it follows that $1/W^2(n,s_{\max})$ is positive increasing. Then proceed to argue that $\frac{n\,(n+1)}{2}-\frac{4\,n^2\,(n+1)\,(n+2)}{Z(n,s_{\max})}+1$ is increasing and is positive in all its domain. To express its derivative, let us define the following function:

$$
G(n,s_{\max}) \;=\; -n+2\,\mathrm{s}_{\max}+\frac{\sqrt{36\left(\frac{n\,(n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-24\,n^2\ (n+1)\ (n+2)}}{3}-n^2-2
$$

The derivative is written as:

$$
\begin{array}{l}
n+\dfrac{4\,n^2\ (n+1)}{G(n,s_{\max})}+\dfrac{4\,n^2\ (n+2)}{G(n,s_{\max})}+\dfrac{8\,n\ (n+1)\ (n+2)}{G(n,s_{\max})}\\
+\dfrac{4\,n^2\ (n+1)\ (n+2)\ \left(2\,n+\frac{24\,n^2\,(n+1)-72\left(n+\frac{1}{2}\right)\left(\frac{n\,(n+1)}{2}-\mathrm{smax}+1\right)+24\,n^2\,(n+2)+48\,n\,(n+1)\,(n+2)}{6\,\sqrt{36\left(\frac{n\,(n+1)}{2}-\mathrm{smax}+1\right)^2-24\,n^2\,(n+1)\,(n+2)}}+1\right)}{G(n,s_{\max})^2}+\dfrac{1}{2}
\end{array}
$$

Which is positive for $n \geq 1$. Now we will argue that $n^3\ \left(\frac{\frac{n}{2}+\frac{\sqrt{3}\,V}{2\,\sqrt{Y}}+\frac{1}{4}}{n}-\frac{\frac{n}{4}-\frac{s_{\max}}{2}-\frac{\sqrt{3\,Y}}{12}+\frac{n^2}{4}+\frac{1}{2}}{n^2}\right)$ is increasing and positive in all the domain. Let us define the following functions:

$$
M(n,s_{\max}) \;=\; \frac{n}{6}+\mathrm{s}_{\max}+2\,n\,\mathrm{s}_{\max}+\frac{9\,n^2}{2}+\frac{5\,n^3}{3}-1,
$$

$$
K(n,s_{\max}) \;=\; 12\left(\frac{n\ (n+1)}{2}-\mathrm{s}_{\max}+1\right)^2-8\,n^2\ (n+1)\ (n+2)\,.
$$

Then the numerator of the derivative of $n^3\ \left(\frac{\frac{n}{2}+\frac{\sqrt{3}\,V}{2\,\sqrt{Y}}+\frac{1}{4}}{n}-\frac{\frac{n}{4}-\frac{s_{\max}}{2}-\frac{\sqrt{3\,Y}}{12}+\frac{n^2}{4}+\frac{1}{2}}{n^2}\right)$ respect to n can be written as:

$$
6\,K(n,s_{\max})\,\mathrm{s}_{\max}-6\,K(n,s_{\max})+\sqrt{3}\,K(n,s_{\max})^{\frac{3}{2}}+9\,K(n,s_{\max})\,n^2
$$

$$
\begin{aligned}
&27\sqrt{3}\sqrt{K(n,s_{\max})}\,n^3+20\sqrt{3}\sqrt{K(n,s_{\max})}\,n^4+6\sqrt{3}\sqrt{K(n,s_{\max})}\,n\\
&+36\sqrt{3}\,M(n,s_{\max})\,n^2\,\mathrm{s}_{\max}\sqrt{\frac{1}{K(n,s_{\max})}}-6\sqrt{3}\sqrt{K(n,s_{\max})}\,n\,\mathrm{s}_{\max}\\
&+162\sqrt{3}\,M(n,s_{\max})\,n^4\sqrt{\frac{1}{K(n,s_{\max})}}-36\sqrt{3}\,M(n,s_{\max})\,n^2\sqrt{\frac{1}{K(n,s_{\max})}}\\
&+6\sqrt{3}\,M(n,s_{\max})\,n^3\sqrt{\frac{1}{K(n,s_{\max})}}+60\sqrt{3}\,M(n,s_{\max})\,n^5\sqrt{\frac{1}{K(n,s_{\max})}}\\
&+72\sqrt{3}\,M(n,s_{\max})\,n^3\,\mathrm{s}_{\max}\sqrt{\frac{1}{K(n,s_{\max})}}+12\sqrt{3}\,K(n,s_{\max})\,M(n,s_{\max})\,n\sqrt{\frac{1}{K(n,s_{\max})}}
\end{aligned}
$$

and the denominator can be written as:

$$12\,K(n,s_{\max}).$$

Given that $K(n,s_{\max})$ is positive for any n in the domain of $h(n)$, it follows that $n^3\cdot\left(\frac{\frac{n}{2}+\frac{\sqrt{3}\,V}{2\sqrt{Y}}+\frac{1}{4}}{n}-\frac{\frac{n}{4}-\frac{s_{\max}}{2}-\frac{\sqrt{3}\,Y}{12}+\frac{n^2}{4}+}{n^2}\right.$ is increasing in n. This expression evaluate in $n=1$ is $\frac{\mathrm{smax}}{2}+\frac{\sqrt{\mathrm{smax}\,(\mathrm{smax}-4)}}{2}+\frac{\left(3\,\mathrm{smax}+\frac{16}{3}\right)\sqrt{\frac{1}{\mathrm{smax}\,(\mathrm{smax}-4)}}}{4}-\frac{1}{4}\geq 0$. We can conclude that $g(n)-n\,g'(n)$ is decreasing in all the domain of $h(n)$. Now we would like to expose a value of n for which $g(n)-n\,g'(n)$ is negative. Let us define $\hat{n}$ as the solution of the following equation:

$$\left(s_{\max}-\frac{(\hat{n}+1)\cdot\hat{n}}{2}-1\right)^2 \;=\; \frac{2\cdot\hat{n}^2\cdot(\hat{n}+1)\cdot(\hat{n}+2)}{3}$$

From definition of $g(n)$ it follows that $g(n)-n\,g'(n)$ can be written as:

$$
\begin{aligned}
g(n)-n\,g'(n) &= \left(1+\frac{Q(n)}{U(n)}\right)-n\cdot\frac{Q'(n)\,U(n)-Q(n)\,U'(n)}{U^2(n)}\\
&= \frac{U^2(n)+Q(n)\,U(n)-n\,Q'(n)\,U(n)+n\,Q(n)\,U'(n)}{U^2(n)}
\end{aligned}
$$

We claim that $h'(\hat{n}-\epsilon)$, ϵ is an arbitrary small positive value, is negative. To prove is enough to argue that:

$$U^2(\hat{n}-\epsilon)+Q(\hat{n}-\epsilon)\,U(\hat{n}-\epsilon)-(\hat{n}-\epsilon)\,Q'(\hat{n}-\epsilon)\,U(\hat{n}-\epsilon)+(\hat{n}-\epsilon)\,Q(\hat{n}-\epsilon)\,U'(\hat{n}-\epsilon)<0.$$

We will argue that $U'(\hat{n}\epsilon)$ is an arbitrary large negative number. Let us define $P(n,s_{\max})$ as:

$$P(n,s_{\max}) \;=\; \left(\frac{n\;(n+1)}{2}-\mathrm{smax}+1\right)^2-\frac{2\,n^2\;(n+1)\;(n+2)}{3}$$

From definition of $U(n)$ it follows that $U'(n)$ is:

$$U'(n) \;=\; -\frac{\mathrm{s}_{\max}-\frac{n\,(n+1)}{2}+\sqrt{P(n,s_{\max})}-1}{2\,n^2}$$

$$-\frac{n+\frac{\frac{2\,n^2\,(n+1)}{3}-2\left(n+\frac{1}{2}\right)\left(\frac{n\,(n+1)}{2}-s_{\max}+1\right)+\frac{2\,n^2\,(n+2)}{3}+\frac{4\,n\,(n+1)\,(n+2)}{3}}{2\sqrt{P(n,s_{\max})}}+\frac{1}{2}}{2\,n}$$

By continuity to the left of $P(n, s_{\max})$ and definition of $\hat{n}$ it follows that $P(\hat{n}-\epsilon, s_{\max})$ is arbitrarily close to 0. Then it follows that $U'(\hat{n}-\epsilon)$ is an arbitrarily large negative number. From definition of $Q(n)$, $Q(n)$ is an increasing function, then Q'(n)¿0, additionally $U(n)$ and $Q(n)$ are positive, in all the domain of $h(n)$. Then we can conclude that $h'(\hat{n}-\epsilon) < 0$. Now we proceed to maximize function $h(n)$ when $s_{\max}$ is large. Let $n = s_{\max}^{\frac{1}{3}} + \epsilon$. We proceed to compute $h(s_{\max}^{\frac{1}{3}+\epsilon})$ for $\epsilon \in [-\delta, \delta]$.

$$\begin{aligned}
U(s_{\max}^{\frac{1}{3}+\epsilon}) &= \frac{\left(s_{\max}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}\cdot(s_{\max}^{\frac{1}{3}+\epsilon}+1)}{2}-1\right)}{2\cdot s_{\max}^{\frac{1}{3}+\epsilon}} \\
&\quad + \frac{\sqrt{\left(s_{\max}-\frac{(s_{\max}^{\frac{1}{3}+\epsilon}+1)\cdot s_{\max}^{\frac{1}{3}+\epsilon}}{2}-1\right)^2-4\cdot s_{\max}^{\frac{1}{3}+\epsilon}\cdot\frac{s_{\max}^{\frac{1}{3}+\epsilon}\cdot(s_{\max}^{\frac{1}{3}+\epsilon}+1)\cdot(s_{\max}^{\frac{1}{3}+\epsilon}+2)}{6}}}{2\cdot s_{\max}^{\frac{1}{3}+\epsilon}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}-\frac{s_{\max}^{-\frac{1}{3}-\epsilon}}{2}+\sqrt{\left(\frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}-\frac{s_{\max}^{\frac{-1}{3}-\epsilon}}{2}\right)^2-\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{6}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\sqrt{\left(\frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}\right)^2-\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{6}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\sqrt{\frac{s_{\max}^{\frac{4}{3}-2\,\epsilon}}{4}-\frac{s_{\max}}{4}+\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{16}-\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{6}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\sqrt{\frac{s_{\max}^{\frac{4}{3}-2\,\epsilon}}{4}-\frac{s_{\max}}{4}-\frac{5\,s_{\max}^{\frac{2}{3}+2\,\epsilon}}{48}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2} \\
&= s_{\max}^{\frac{2}{3}-\epsilon}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}
\end{aligned}$$

$$\begin{aligned}
Q(s_{\max}^{\frac{1}{3}+\epsilon}) &= 1+\frac{s_{\max}^{\frac{1}{3}+\epsilon}\,(s_{\max}^{\frac{1}{3}+\epsilon}+1)}{2}+\frac{s_{\max}^{\frac{1}{3}+\epsilon}\,(s_{\max}^{\frac{1}{3}+\epsilon}+1)\,(s_{\max}^{\frac{1}{3}+\epsilon}+2)}{\frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\sqrt{\frac{s_{\max}^{\frac{4}{3}-2\,\epsilon}}{4}-\frac{s_{\max}}{4}-\frac{5\,s_{\max}^{\frac{2}{3}+2\,\epsilon}}{48}}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{2}+\frac{s_{\max}^{1+3\,\epsilon}}{\frac{s_{\max}^{\frac{2}{3}-\epsilon}}{2}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}+\sqrt{\frac{s_{\max}^{\frac{4}{3}-2\,\epsilon}}{4}-\frac{s_{\max}}{4}-\frac{5\,s_{\max}^{\frac{2}{3}+2\,\epsilon}}{48}}} \\
&\approx \frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{2}+\frac{s_{\max}^{1+3\,\epsilon}}{s_{\max}^{\frac{2}{3}-\epsilon}-\frac{s_{\max}^{\frac{1}{3}+\epsilon}}{4}}
\end{aligned}$$

$$\begin{aligned} h(s_{\max}^{\frac{1}{3}+\epsilon}) &= \frac{2 \cdot s_{\max}^{\frac{1}{3}+\epsilon}}{1 + \frac{Q(s_{\max}^{\frac{1}{3}+\epsilon}, s_{\max})}{U(s_{\max}^{\frac{1}{3}+\epsilon}, s_{\max})}} \\ &= \frac{2\, s_{\max}^{\frac{1}{3}}}{s_{\max}^{-\epsilon} + s_{\max}^{-\epsilon} \left[\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{2\, s_{\max}^{\frac{2}{3}-\epsilon} - \frac{s_{\max}^{\frac{1}{3}+\epsilon}}{2}} + \frac{s_{\max}^{1+3\,\epsilon}}{s_{\max}^{\frac{4}{3}-2\,\epsilon} - \frac{s_{\max}}{2} + \frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{4}} \right]} \end{aligned}$$

Let us denote as $dh(s_{\max}^{\frac{1}{3}+\epsilon})$ the denominator of $h(s_{\max}^{\frac{1}{3}+\epsilon})$. Maximizing $h(s_{\max}^{\frac{1}{3}+\epsilon})$ is equivalent to minimizing $dh(s_{\max}^{\frac{1}{3}+\epsilon})$.

$$\begin{aligned} dh(s_{\max}^{\frac{1}{3}+\epsilon}) &= s_{\max}^{-\epsilon} + s_{\max}^{-\epsilon} \left[\frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{2\, s_{\max}^{\frac{2}{3}-\epsilon} - \frac{s_{\max}^{\frac{1}{3}+\epsilon}}{2}} + \frac{s_{\max}^{1+3\,\epsilon}}{s_{\max}^{\frac{4}{3}-2\,\epsilon} - \frac{s_{\max}}{2} + \frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{4}} \right] \\ &= s_{\max}^{\epsilon} + \frac{s_{\max}^{\frac{2}{3}+\epsilon}}{2\, s_{\max}^{\frac{2}{3}-\epsilon} - s_{\max}^{\frac{1}{3}+\epsilon}} + \frac{s_{\max}^{1+2\,\epsilon}}{s_{\max}^{\frac{4}{3}-2\,\epsilon} - \frac{s_{\max}}{2} + \frac{s_{\max}^{\frac{2}{3}+2\,\epsilon}}{4}} \\ &= s_{\max}^{-\epsilon} + \frac{1}{2\, s_{\max}^{-2\,\epsilon} - \frac{s_{\max}^{-\frac{1}{3}}}{2}} + \frac{1}{s_{\max}^{\frac{1}{3}} - \frac{s_{\max}^{-2\,\epsilon}}{2} + \frac{s_{\max}^{-\frac{1}{3}}}{4}} \end{aligned}$$

$$\begin{aligned} dh(s_{\max}^{\frac{1}{3}+\epsilon})' &= -s_{\max}^{-\epsilon} \cdot \log s_{\max} + \frac{4\, s_{\max}^{-2\,\epsilon} \log s_{\max}}{\left(2\, s_{\max}^{-2\,\epsilon} - \frac{s_{\max}^{-\frac{1}{3}}}{2}\right)^2} + \frac{2\, s_{\max}^{-2\,\epsilon} \log s_{\max}}{\left(s_{\max}^{\frac{1}{3}} - \frac{s_{\max}^{-2\,\epsilon}}{2} + \frac{s_{\max}^{-\frac{1}{3}}}{4}\right)^2} \\ &\approx -s_{\max}^{-\epsilon} \cdot \log s_{\max} + \frac{4\, s_{\max}^{-2\,\epsilon} \log s_{\max}}{\left(2\, s_{\max}^{-2\,\epsilon} - \frac{s_{\max}^{-\frac{1}{3}}}{2}\right)^2} \\ &= 0 \end{aligned}$$

Then we need to solve the following equation:

$$\begin{aligned} -s_{\max}^{-\epsilon} + \frac{4\, s_{\max}^{-2\,\epsilon}}{\left(2\, s_{\max}^{-2\,\epsilon} - \frac{s_{\max}^{-\frac{1}{3}}}{2}\right)^2} &= 0 \\ 2\, s_{\max}^{\frac{-\epsilon}{2}} &= 2\, s_{\max}^{-2\,\epsilon} - \frac{s_{\max}^{-\frac{1}{3}}}{2} \end{aligned}$$

$$2\, s_{\max}^{\frac{-\epsilon}{2}} \approx 2\, s_{\max}^{-2\,\epsilon}$$

Thus $\epsilon = 0$. Then it follows that for $s_{\max}$ large, $n^* = s_{\max}^{\frac{1}{3}}$ maximizes $h(n)$. □